TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

TAINA PELTOLA

SEIKKAILUPELIEN SUUNNITTELUTYÖKALUN OMINAISUUKSIEN MÄÄRITTÄMINEN JA SUUNNITELMAN VIENTI PELIMOOTTORIIN

Diplomityö

Tarkastajat:
professori Tommi Mikkonen,
DI Timo Kellomäki
Tarkastajat ja aihe hyväksytty
Tieto- ja sähkötekniikan
tiedekuntaneuvoston kokouksessa
14. tammikuuta 2015

TIIVISTELMÄ

TAINA PELTOLA: Seikkailupelien suunnittelutyökalun ominaisuuksien määrittäminen ja suunnitelman vienti pelimoottoriin
Tampereen teknillinen yliopisto
Diplomityö, 67 sivua, 8 liitesivua
Syyskuu 2015
Tietotekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Ohjelmistotuotanto
Tarkastajat: professori Tommi Mikkonen, DI Timo Kellomäki

Avainsanat: seikkailupelit, pelisuunnittelu, suunnittelutyökalu, pelimoottori

Seikkailupelit ovat tarinallisia tietokonepelejä, joissa pelaaja ratkoo tehtäviä juonen etenemiseksi. Seikkailupelien suunnitteluun ei ole tarjolla siihen tarkoitettuja työkaluja. Seikkailupelejä suunnitellaan esimerkiksi paperilla ja kynällä, tekstinkäsittelyohjelmalla tai graafinpiirto-ohjelmalla. Paperille tehdyn suunnitelman muokkaaminen ja jakaminen on vaivalloista ja suunnitelma voi olla epäjärjestyksessä. Tekstinkäsittelyohjelmassa tulee itse luoda rakenne ja graafien piirto voi tapahtua eri ohjelmassa. Seikkailupelien toteuttamiseen sen sijaan on tarjolla useita pelimoottoreita. Suunnitelmassa kehitellyt peliobjektit, kuten tavarat ja huoneet, joudutaan lisäämään pelimoottoriin manuaalisesti yksi kerrallaan.

Tässä diplomityössä tutkitaan, mitä ominaisuuksia tarvitaan työkaluun, jolla voidaan suunnitella seikkailupelejä. Työssä kerätään menetelmiä ja ohjeita, joiden avulla seikkailupelejä suunnitellaan. Näiden kautta määritellään työkalun ominaisuudet. Lisäksi tutkitaan, miten suunnitteluvaiheessa tehty työ saataisiin hyödynnettyä seikkailupelin toteutuksen pohjan luomisessa. Tätä varten työkaluun lisätään ominaisuus, joka luo geneerisen XML-tiedoston suunnitelmasta. Siihen tallennetaan suunnitelmassa luodut objektit. Pelinkehittäjien tulee luoda komponentti, joka lukee tiedon heidän käyttämäänsä pelimoottoriin. XML:n vientiä Visionaire-pelimoottoriin tutkitaan työssä alustavasti.

Työssä esitellään työkalun prototyyppi, joka toteutettiin ominaisuuksien pohjalta. Prototyyppi toteutettiin itsenäisesti ilman rahoitusta. Toteutetun työkalun ominaisuuksia arvioidaan työssä määritellyn testipelin kautta. Suunnittelutyökalu tarjoaa oleelliset osat seikkailupelien suunnitteluun: tarinan ja juonen, tehtävien ja niiden riippuvuuksien ja kartan suunnittelu. Se tukee suunnitteluohjeiden käyttöä, mutta ei aina ohjaa siihen. Työkalun ansiosta suunnitelma on jäsennelty selkeään rakenteeseen. Toisaalta siihen ei voida esimerkiksi hahmotella suoraan piirrosta, kuten muistikirjaan voidaan. Työkalu tarvitsee lisää ominaisuuksia, jotta XML:n vienti pelimoottoriin olisi mahdollista: Visionare-pelimoottorin tavarat ovat joko huoneen tai tavaraluettelon tavaroita. Työkalun prototyypin kautta pystytään viemään vain tavaraluettelon tavaroita, sillä tavaroille ei prototyyppiversiossa pystytä määrittämään sijaintia tietyssä huoneessa. XML:n vientiä pelimoottoreihin tulee tutkia lisää, jotta voidaan varmistaa, että työkalun generoima XML-tiedosto tarjoaa kaiken tarvittavan tiedon. Työkalu vaatii muutenkin jatkokehitystä, koska kaikkia välttämättömiä ominaisuuksia ei ole toteutettu tai viimeistelty. Jatkokehitysideat esitetään työssä. Lisäksi tarvitaan palautetta seikkailupelisuunnittelijoilta, jotta työkalua voidaan arvioida luotettavammin ja priorisoida jatkokehitysideoita.

ABSTRACT

TAINA PELTOLA: Features in a Design Tool for Adventure Games and Exporting the Design for a Game Engine
Tampere University of Technology
Master of Science Thesis, 67 pages, 8 Appendix pages
September 2015
Master's Degree Programme in Information Technology
Major: Software Engineering
Examiner: Professor Tommi Mikkonen, M.Sc. Timo Kellomäki

Keywords: adventure games, game development, design tool, game engine

Adventure games are video games with a story. The player solves puzzles to advance the plot. There are no tools designed especially for designing adventure games. For example, the games are designed with a pen and a paper, a word processor or a graph drawing software. It is complicated to edit and share the design made on paper, and it can be disorganized. With a word processor, the structure needs to be created and drawing the graphs might be done in a different program. Nevertheless, there are multiple game engines available to implement adventure games. The designed objects, such as items and rooms, need to be added manually one by one to the engine.

The thesis studies what kind of features are needed in an adventure game design tool. Methods and guidelines for adventure game design are introduced. These are used to define the features needed in the design tool. In addition, the thesis studies how the design of the game could be used to generate a base for the game in a game engine. The tool generates an XML file where the designed game objects are saved. The game developers need to create a component to import the data from the XML to the game engine they use. Importing the XML to a game engine called Visionaire is studied tentatively in the thesis.

The thesis presents a prototype version of the tool which was implemented according to the defined features. The tool was developed independently without funding. The features are evaluated by designing a test game with the tool. The adventure game design tool has the necessary features to design an adventure game: designing the story and the plot, the puzzles and their dependencies, and the map. The tool supports the use of the guidelines but does not always guide the user to use them. With the tool the design of the game is structured and in clear order. On the other hand you cannot for example sketch a drawing directly with it as you could on paper. The tool needs more features to make the import fully possible: The items in Visionaire are either items in a room or in the inventory. In the prototype of the tool only inventory items can be imported as you cannot set a room location for the item. Importing the XML to game engines needs to be studied more to ensure that all the needed data is available in the generated XML. Further development is needed to finalize the prototype and implement more features. The development ideas are presented in the thesis. Also, feedback from adventure game designers is needed so the tool can be evaluated more solidly and the development ideas can be prioritized.

ALKUSANAT

Halusin tehdä diplomityöni minua erityisesti kiinnostavasta aiheesta. Aiheen valitseminen vei kauan, mutta lopulta päädyin tähän omaan aiheeseen liittyen seikkailupeleihin. Seikkailupelit ovat olleet osa elämääni niin kauan kuin muistan. Vaikka tähän asti olen vain pelannut niitä, ehkä vielä otan osaa niiden kehittämiseenkin.

Kiitän työn ohjaajia ja tarkastajia Timo Kellomäkeä ja Tommi Mikkosta ohjauksesta ja hyvistä kommenteista. Kiitos puolisolleni Petrille siitä, että jaksoit kannustaa, antaa kommentteja ja istua päivästä toiseen ääneenkin tuskailevan diplomityönkirjoittajan kanssa samassa huoneessa. Oikoluvusta, kommenteista ja kannustuksesta kiitän myös veljiäni Olavia ja Simoa. Haluan kiittää myös seikkailupelisuunnittelija Dave Gilbertiä, joka vastasi sähköpostilla laittamiini kysymyksiin seikkailupelien suunnittelusta, sekä Ken Williamsia, joka vastasi kysymykseeni kuvien käytöstä. Mukavaa, että alalta löytyy avuliaita ihmisiä! Mikäli tähän työhön liittyen herää kysymyksiä, minuun voidaan ottaa yhteyttä alla olevaan sähköpostiosoitteeseen.

Tampereella, 22.8.2015

Taina Peltola

tainanposti@gmail.com

SISÄLLYSLUETTELO

TERMIT JA LYHENTEET

Adventure Game Studio	Pelimoottori, joka on tarkoitettu point-and-click-seikkailupelien tekemiseen.
AGS	Lyhenne: Adventure Game Studio.
Casual-pelaaja	Pelaaja, joka pelaa satunnaisesti viihteen vuoksi.
Hotspot-kohde/alue	Asia tai tavara ruudulla, jota pelaaja ei voi kerätä tavaraluetteloonsa, mutta voi muuten käyttää tai tutkia ruudulla.
Immersio	”Uppoutuminen” pelin maailmaan ja eläytyminen pelin sisältöön.
Indiekehittäjä	Independent developer, riippumaton kehittäjä.
Kontekstisidonnainen kuva	Ruutu näytetään tietystä kulmasta eikä kamera seuraa pelaajaa.
Lateraalinen ajattelu	Epäsuora ja luova päättelykyky.
Maailmankartta	Seikkailupelissä karttanäkymä, jossa esitetään lokaatioita, joihin pelaajahahmo voi mennä
NPC-hahmo	Non-player-character. Hahmo, jota pelaaja ei ohjaa.
Ongelma	Engl. problem. Seikkailupelin tehtävän osa, joka kuvaa tavoitteen ja tilanteen, johon on löydettävä ratkaisu.
Pelaajahahmo	Hahmo, jota pelaaja ohjaa.
Point-and-click-peli	Osoita-ja-klikkaa-peli. Seikkailupeli, jossa käytetään objekteja klikkaamalla niitä ruudulla.
Pullonkaula tehtävägraafissa	Tilanne, jossa kaikki tehtävät yhdistyvät yhteen samaan tehtävään.
Pulmapeli	Logiikkatehtävä, joka on yleensä esitetty yhdellä ruudulla.
Puzzle Dependency Chart	Ron Gilbertin luoma työkalu seikkailupelien tehtävien riippuvuuksien suunnitteluun.
Ratkaisu	Seikkailupelin tehtävän osa, joka kuvaa, miten ongelma selvitetään.
Remake/Remastered	Uusi versio pelistä.
Script Creation Utility for Maniac Mansion	Komentosarjakieli, jolla on tehty useita Lucasfilm Gamesin/LucasArtsin pelejä.

SCUMM	Lyhenne: Script Creation Utility for Maniac Mansion.
Tavaraluettelo	Engl. inventory. Pelaaja voi kerätä tavaraluetteloon tavaroita, joita pelaajahahmo kantaa mukanaan.
Tehtävä	Seikkailupelissä oleva tavoite, joka koostuu ongelmasta ja ratkaisusta.
Toimintaelementti	Toimintapainotteisia elementtejä pelissä, kuten ampuminen.
Toimintaseikkailu	Engl. action-adventure game. Peli, jossa on seikkailupelille ominaisia piirteitä sekä toimintaelementtejä.
Komentotulkki	Engl. text parser. Tekstiseikkailupelin osa, joka tulkitsee pelaajan syöttämiä sanoja komennoiksi.
Tekstiseikkailupeli	Seikkailupeli, jossa ei ole grafiikoita, vaan syötteet ja tulosteet annetaan tekstinä.
Tutkimuspeli	Engl. exploration game. Peli, jossa keskitytään pelin maailman tutkimiseen.
Visual novel	Peli, jossa tarina esitetään tekstinä ja sitä elävöitetään taustakuvilla ja animaatioilla sekä musiikilla ja äänitehosteilla.
Välianimaatio	Engl. cut scene. Pelattavien osuuksien välissä, tai pelin alussa tai lopussa oleva animaatio, joka esittelee tarinaa ja kuljettaa juonta.

1. JOHDANTO

Tietokoneille on tehty seikkailupelejä jo neljäkymmentä vuotta. Ne ovat tarinavetoisia pelejä, joissa pelaajahahmo tutkii maailmaa ja ratkoo tehtäviä. Seikkailupelien tarinan ja tehtävien suunnittelumenetelmät ovat muotoutuneet pelejä tehtäessä. Kokeneet seikkailupelisuunnittelijat ovat kirjoittaneet ohjeita, jotta peleissä ei toistettaisi huonoksi havaittuja keinoja. Kuitenkaan ohjeita ei ole tarkoitus noudattaa tarkasti, vaan ne auttavat suunnittelussa. Seikkailupelien teko vaatii luovuutta tarinan ja mielenkiintoisten tehtävien luomiseksi.

Seikkailupelien toteuttamiseen on olemassa valmiita pelimoottoreita työkaluineen. Kuitenkin jo suunnitteluvaiheessa voitaisiin hyötyä työkalusta. Seikkailupelejä suunnitellaan esimerkiksi paperilla ja kynällä, tekstinkäsittelyohjelmalla tai graafinpiirto-ohjelmalla. Paperille tehdyn suunnitelman muokkaaminen ja jakaminen on vaivalloista. Suunnitelma saattaa myös olla epäjärjestyksessä, kun ideoita on kirjoitettu sinne tänne. Tekstinkäsittelyohjelmassa ei ole valmiina minkäänlaista rakennetta seikkailupelin suunnitelmalle, vaan se tulee luoda itse. Graafit saatetaan piirtää eri ohjelmassa, jolloin suunnittelija joutuu vaihtelemaan ohjelmien välillä ja tekstisuunnitelma ja graafit ovat irrallisia toisistaan. Seikkailupelin suunnittelijat hyötyisivät ensisijaisesti suunnittelutyökalun käytöstä. Siitä voisivat hyötyä myös muut pelin kehittämiseen osallistuvat henkilöt, jotka tarvitsevat kehityksessä erilaisia tietoja suunnitelmasta. Suunnittelutyökalun teossa olisi huomioitava, millaisia seikkailupelejä nykyään tehdään, ja trendit, joiden suuntaan ollaan menossa. Ensimmäinen tutkimuskysymys on: Minkälainen työkalu ja ominaisuudet ovat hyödyllisiä seikkailupelin suunnitteluvaiheessa?

Seikkailupelien työkalujen tavoitteena on tehostaa pelin kehitystä. Pelin toteuttamiseen käytetään markkinoilla olevaa tai peliyhtiön omaa pelimoottoria. Pelimoottoriin syötetään manuaalisesti paljon dataa, joka on suunnitteluvaiheessa jo määritelty, joten suunnittelutyökalusta tulisi olla yhteys pelimoottoriin. Tästä seuraa toinen tutkimuskysymys: Miten suunnitteluvaiheessa tehty työ saataisiin hyödynnettyä seikkailupelin toteutuksen pohjan luomisessa?

Työssä suunniteltiin ja toteutettiin seikkailupelien suunnittelutyökalun prototyyppi. Diplomityö ja prototyyppi toteutettiin itsenäisenä projektina ilman rahoitusta. Kirjoittaja on pelannut seikkailupelejä lapsesta saakka, mutta ei ole ennen suunnitellut niitä. Pelatut pelit ovat olleet perinteisen tyylisiä seikkailupelejä, joissa tehtävien ratkonnalla on paljon painoa. Jotkut lähteettömät tiedot työssä perustuvat lukuisiin seikkailupeleihin, joita kirjoittaja on pelannut. Kirjoittajalle vähemmän tuttuihin peleihin, kuten uusiin tarinallisiin peleihin, tutustuttiin YouTube-videoiden avulla.

Luvussa 2 esitetään seikkailupelien historia ja kehittyminen tekstiseikkailupeleistä nykytilaan. Lisäksi esitellään nykyisiä kehittäjiä ja pelejä sekä suuntaa, johon seikkailupelien arvellaan olevan menossa. Luvussa 3 käydään läpi seikkailupelien ominaisuuksia sekä menetelmiä, joita voidaan käyttää seikkailupelien suunnitteluun. Näiden pohjalta määritellään ominaisuuksia suunnittelutyökaluun. Lisäksi luvussa määritellään työkalun käyttäjät. Luvussa 4 esitellään seikkailupelien toteuttamiseen käytettyjä pelimoottoreita. Tarkemmin esitellään Adventure Game Studio - ja Visionaire-pelimoottorit sekä niiden käyttämät XML-muodot peliprojektin tallentamiseen. Lopuksi todetaan vastaus toiseen tutkimuskysymykseen. Luvussa 5 esitellään, millainen toteutettu työkalu on, sekä lyhyesti sen toteutus ja ratkaisut toteutuksessa. Luvussa 6 määritellään ensin työkalun testaukseen tarvittava testipeli. Sen jälkeen esitetään, millaisia asioita työkalusta huomattiin, kun testipeliä suunniteltiin työkalulla ja pohditaan, mitä asioita pitäisi korjata ja lisätä. Lisäksi tutkitaan, kuinka helposti XML:n vienti pelimoottoriin onnistuu. Lopuksi esitellään uusia ominaisuuksia, joita työkalun seuraavaan versioon kannattaa lisätä. Luvussa 7 käydään läpi, mitä tutkittiin ja mihin tuloksiin päädyttiin. Lisäksi mietitään, mitkä ovat seuraavat vaiheet työkalun kehittämisessä.

2. SEIKKAILUPELIEN HISTORIA JA NYKYTILA

Seikkailupeleillä tarkoitetaan tässä työssä tietokoneella, konsolilla tai mobiililaitteella pelattavia elektronisia seikkailupelejä. Tässä luvussa kuvataan, miten seikkailupelit ovat syntyneet ja muovautuneet nykyiseen muotoonsa ja mihin suuntaan ne jatkavat kehittymistään. Työkalun suunnittelussa on otettava huomioon, millaisia seikkailupelejä tehdään nykyisin ja lähitulevaisuudessa.

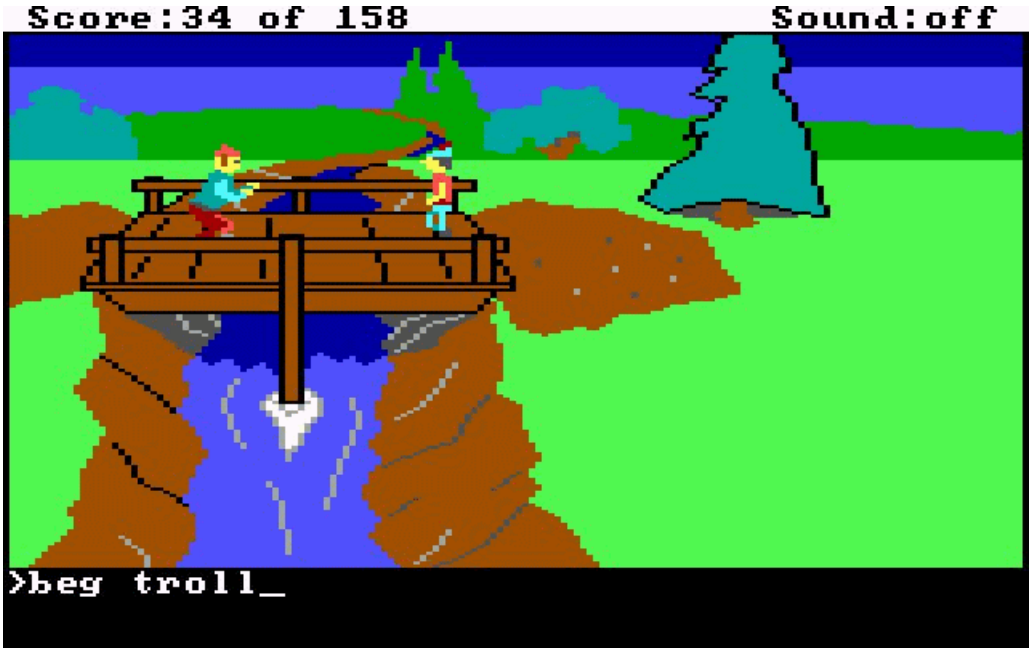
2.1 Historia

Ensimmäinen seikkailupeli oli tekstiseikkailupeli nimeltään *Colossal Cave Adventure* (tai *Adventure*) 1970-luvun puolivälistä. Tekstiseikkailu on peli, jossa tarina ja maailma esitetään tekstinä ja pelaaja kirjoittaa, mitä haluaa tehdä. Esimerkiksi jos pelaaja haluaa liikkua pohjoisessa olevaan ruutuun, hän kirjoittaa ”n” niin kuin ”north”. Teksti-seikkailupelejä tehdään edelleen, mutta ei kaupallisesti. [1]

Ensimmäinen seikkailupeli, jossa oli grafiikat mukana, oli Roberta ja Ken Williamsin perustaman On-Line Systems -pelistudion (myöhemmin Sierra On-Line ja Sierra Entertainment [2]) *Mystery House* [3, s. 135]. Se julkaistiin vuonna 1980 [3, s. 135]. Kuten kuvasta 2.1 näkyy, grafiikat olivat vain ääriviivoja mustalla taustalla, ja syötteet annettiin edelleen pelkästään kirjoittamalla. Kuitenkin pelaaja näki, millaisessa paikassa hän on, mitä tavaroita siellä on ja minne siitä pääsisi liikkumaan. Pelaaja näki myös helposti paikan tilan: Esimerkiksi kun pelaaja avaa suljetun oven, kuva piirretään uudelleen ovi avoimena. [4] [5]

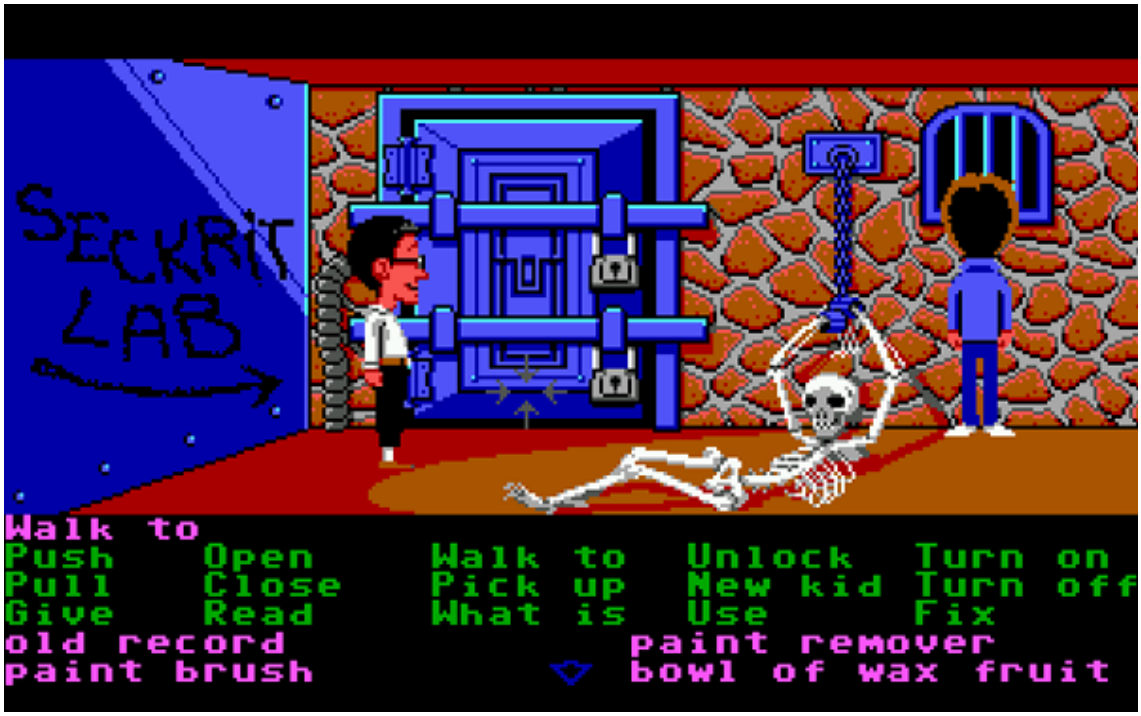
Kuva 2.1: Kuvakaappaus On-Line Systemsin pelistä Mystery House. Kuva otettu lähteestä [6].

Ensimmäinen graafinen seikkailupeli, jossa oli ruudulla liikuteltava pelaajahahmo, oli Sierra On-Linen peli *King's Quest* vuodelta 1984 [1, s. 84]. Kuvassa 2.2 näkyy oikealla pelaajahahmo ja vasemmalla trolli. Pelaaja on syöttämässä komentoa ”beg troll”. Aikaisemmin seikkailupeleissä ei ole ollut pelaajahahmoa, vaan ne ovat olleet ensimmäisestä persoonasta kuvattuja. Tämän myötä jouduttiin miettimään tarkemmin, kuka pelaajahahmo on, ja myös tarinasta tuli sitä kautta selkeämpi. Muu kuin pelaajahahmon liikuttaminen tehtiin edelleen tekstisyötteenä. [3, s. 138]



Kuva 2.2: Kuvakaappaus Sierra On-Linen pelistä King's Quest. Ruudulla näkyy pelaajahahmo, jota voi liikuttaa ruudulla. Muut komennot annetaan tekstinä. Kuva otettu lähteestä [7].

Seikkailupelien kultakausi alkoi, kun Ron Gilbert kumppaneineen teki Lucasfilm Games -yhtiölle (myöhemmin osa LucasArts-yhtiötä, nykyään Disneyn omistama [8]) pelin, jossa ei enää kirjoitetakaan komentoja. Ruudulta valitaan haluttu komento klikkaamalla sitä rajatusta komentojen listasta. Sen jälkeen ruudulta valitaan asia, johon komentoa halutaan käyttää. Tämä *point-and-click*-peli (osoita-ja-klikkaa) oli *Maniac Mansion*, joka julkaistiin vuonna 1987. Siinä myös pelataan useampaa hahmoa, jotka pelaaja saa valita vaihtoehdoista pelin alussa [9]. Kuvassa 2.3 näkyy kaksi pelaaja-hahmoa. Komentojen lista on alareunassa vihreällä värillä. Tavaraluettelo (inventory) kertoo, mitä tavaroita pelaajahahmo on kerännyt ja kantaa mukanaan. Se on esitetty vaaleanpunaisena tekstinä alareunassa. Komento valitaan kursorilla, joka näkyy kuvassa oven alareunan kohdalla. Gilbert kirjoitti *Maniac Mansion* -peliä varten Script Creation Utility for Maniac Mansion -komentosarjakielen, lyhennettynä SCUMM. SCUMM-järjestelmällä tehtiin useita yhtiön tunnettuja pelejä, kuten *The Secret of Monkey Island*, *Indiana Jones and the Fate of Atlantis*, *Day of the Tentacle* ja *Sam & Max Hit the Road*. LucasArtsin peleille on tyypillistä niiden huumori, ja siitä ovat ottaneet mallia myös muut seikkailupelinkehittäjät. [3, s. 200–202]



Kuva 2.3: Kuvakaappaus Lucasfilm Gamesin pelistä Maniac Mansion. Suoritettava komento valitaan listasta komentoja (vihreä teksti). Komentojen alla näkyy tavaraluettelo (vaaleanpunainen teksti). Ruudulla näkyy tässä kaksi pelaajahahmoa. Kuva otettu lähteestä [10].

Myös Sierralla kehiteltiin käyttöliittymää, jossa ei tarvitsisi arvailla, mitä sanoja komentotulkki (text parser) odottaa, vaan pystyttäisiin vuorovaikuttamaan asioiden kanssa suoraan kuvasta [4]. Vuonna 1990 julkaistiin Sierran ensimmäinen point-and-click-seikkailupeli *King's Quest V* [4]. Kuvassa 2.4 on kuvakaappaus pelistä. Käytettävissä olevat komennot on esitetty kuvina yläreunassa olevassa valikossa. Laukun kuvasta päästään tarkastelemaan tavaraluetteloa. Laukun vasemmalla puolella on ruutu, johon pelaaja voi valita tavaran, jolloin hän voi käyttää sitä nopeammin kuin valitsemalla sen aina tavaraluettelon kautta.

Vuonna 1993 julkaistiin Cyanin kehittämä peli *Myst* [3, s. 258–260]. Pelissä on aikaansa nähden hieno 3D-maailma [3, s. 260], jota pelaaja tutkii ensimmäisen persoonan perspektiivistä ja ratkoo tehtäviä [11] [12]. Pelaajalla ei ole aluksi selvää tavoitetta tai tarinaa tiedossa, vaan ne selviävät tutkimalla maailmaa [11] [12]. Tehtävät eivät ole samanlaisia kuin edeltäneissä seikkailupeleissä: maailmassa on erilaisia laitteita vipuineen ja nappeineen, joista pitää selvittää, miten niitä käytetään ja esimerkiksi syöttää toisaalta maailmasta löytynyt koodi laitteeseen [11] [12]. Pelissä ei myöskään ole tavaroita, joita pelaaja voisi lisätä tavaraluetteloonsa tai dialogeja muiden hahmojen kanssa [11] [12]. Koska peli keskittyy pitkälti tutkimiseen ja irralliselta tuntuviin logiikka- ja muistitehtäviin, voidaan pohtia, onko se edes seikkailupeli, vaan ennemminkin tutkimuspeli (exploration game) pulmapeleillä.

Kuva 2.4: Kuvakaappaus Sierran pelistä King's Quest V: To Heir is Human. Kun kursori vedetään ruudun yläreunaan, aukeaa komentovalikko. Komennot on esitetty kuvina. Kuva otettu lähteestä [13].

Myst oli useita vuosia kaikkien aikojen parhaiten myynyt tietokonepeli [14, s. 445]. Tosin *Myst* julkaistiin, kun CD-ROM alkoi yleistyä, ja se hyödynsi CD-ROM:in mahdollisuudet, mikä todennäköisesti vaikutti sen menestykseen [3, s. 259]. Seikkailupeligenreä kokeiltiin viedä eteenpäin myös muilla tavoilla, esimerkiksi Sierra julkaisi pelin *Phantasmagoria*, jossa näyttelijät näyttelivät pelin hahmojen liikkeet ja ne nauhoitettiin ja liitettiin peliin [3, s. 142]. Seikkailupelien kultakauden voidaan sanoa tulleen päätökseensä ennen vuosituhannen vaihdetta [15].

2.2 Nykytila

Seikkailupelejä on julkaistu jatkuvasti, vaikka genren on väitetty kuolleen kultakauden jälkeen [14, s. 445] [16]. Tunnettujen julkaisujen määrä on kyllä vähentynyt, ja tehnyt nousua vasta viime vuosina [15]. Rollings ja Adams [14, s. 445–446] arvelevat, että tämä johtuu siitä, että seikkailupelit eivät ole käyttäneet uusinta näyttöteknologiaa (vaikka olisivat voineet) ja eivät siksi ole saaneet niin paljoa mediahuomiota. Seikkailupelit eivät yllä myydyimpien pelien listoille [17] [18] [19]. Genre ei enää myöskään ole erityisemmin kehittynyt pelimekaniikoiltaan – ennen kuin viime vuosina [16]. Viime aikoina on julkaistu seikkailupelejä, jotka ovat myyneet kohtuullisen hyvin, kuten Telltalen kehittämä *The Walking Dead* [20].

2.2.1 Kehittäjät

Nykyisin on useita pelistudioita, jotka kehittävät ja/tai julkaisevat seikkailupelejä. Jotkut ovat keskittyneet puhtaasti seikkailupeleihin, kun toiset taas julkaisevat myös muiden

genrejen pelejä. Taulukossa 2.1 on esitetty merkittäviä seikkailupelistudioita ja heidän seikkailupelejään.

Taulukko 2.1: Seikkailupelikehittäjiä ja -julkaisijoita [21] [22] [23] [24].

Pelistudio	**Pelejä**
Daedalic Entertainment	The Whispered World -sarja, Deponia-sarja, The Night of The Rabbit
Double Fine	Broken Age, Grim Fandango Remastered
Telltale	Tales of Monkey Island, The Walking Dead, Sam & Max
Wadjet Eye Games	Blackwell-sarja, Gemini Rue, Resonance

Saksalainen Daedalic Entertainment -peliyhtiö on julkaissut useita käyttöliittymältään ja tehtäviltään perinteisiä seikkailupelejä. Esimerkiksi *The Whispered World* ja *The Night of the Rabbit* on kuitenkin ulkoisesti tuotu nykypäivään. Niiden grafiikat ovat upeita piirroksia ja ainakin *The Night of the Rabbit* -pelissä on tunnelmaa luovia tehosteita, kuten lumisade. Kuvassa 2.5 on kuvakaappaus pelistä. Oikealla kuvassa näkyy tehosteena lumisadetta.

Kuva 2.5: Kuvakaappaus Daedalic Entertainmentin pelistä The Night of the Rabbit. Oikealla näkyy lumisadetehoste. Kuva otettu suoraan pelistä.

Seikkailupelejä tehdään niin ilman budjettia harrasteprojekteina kuin miljoonaluokan projekteina. Esimerkiksi tarinaan ja valintoihin keskittyvän *Heavy Rain* -pelin (2010) budjetti oli kaikkineen 40 miljoonaa euroa, josta 16,7 miljoonaa oli kehitykseen [25]. Seikkailupelejä kehitetään myös joukkorahoituksen avulla. Esimerkiksi seikkailupeli-

suunnittelijaveteraanit Ron Gilbert ja Gary Winnick keräsivät peliin *Thimbleweed Park* yli 600 000 dollaria [26]. Se on tehty vanhojen LucasArts-pelien hengessä [26]. Double Fine -pelistudio keräsi *Brogen Age* -pelille 3,3 miljoonaa dollaria [27]. Pelisuunnittelijana toimii seikkailupelisuunnittelijaveteraani Tim Schafer, joka on suunnitellut suositun *Monkey Island* -sarjan [27]. Myös vähemmän tunnettuja nimiä rahoitetaan, mutta ei yleensä yhtä suurilla summilla [28].

Nykyään seikkailupelejä tehdään ja myydään myös erillisinä jaksoina (episode, part tai act). Sama tarina jatkuu, mutta jaksot ilmestyvät yksitellen. Tämä on hyödyllistä pelinkehittäjien kannalta siksi, että jakso saadaan myyntiin aiemmin kuin kokonainen peli saataisiin. Tällöin saadaan rahaa seuraavan jakson kehittämiseen, eikä riskikään ole niin iso kuin silloin kun tehdään kerrallaan täyspitkä seikkailupeli. Pelaaja joutuu odottelemaan seuraavan jakson ilmestymistä, jos hän pelaa peliä tuoreeltaan, mutta toisaalta hän voi ostaa halvemmalla jakson kuin kokonaisen pelin ja kokeilla sitä, ennen kuin päättää, haluaako edes ostaa seuraavia osia.

2.2.2 Suunta

Dave Gilbert Wadjet Eye Games -pelistudiolta uskoo, että nykyään trendinä on keskittyä immersioon ja maailman tutkimiseen. Immersio tarkoittaa, että pelaaja ”uppoutuu” pelin maailmaan ja eläytyy peliin. Pelikokemuksesta itsestään tehdään nautinnollisempi ja hauskempi verrattuna perinteiseen tehtävien ratkomiseen, jossa tehtävät voivat olla niin vaikeita, että niihin jää jumiin. [29]

Myös Ron Gilbert sanoo, että nykyään on enemmän casual-pelaajia, jotka pitävät pelejä kokemuksena, samalla lailla kuin elokuvien katsominen. He eivät halua epäonnistua jatkuvasti ja yrittää ratkaista tehtävää monta päivää. Gilbert huomioi tämän peleissään ja ei tee enää niin vaikeita tehtäviä kuin ennen. Pelaajat, jotka pitävät todella hankalista tehtävistä, ovat Gilbertin mukaan niche-ryhmä. [30]

Esimerkiksi *The Walking Dead* keskittyy enemmän interaktiiviseen tarinankerrontaan kuin tehtävien ratkomiseen. Lisäksi siinä on haarautuva juoni, mikä voi tuoda uutta mielenkiintoa seikkailupeleihin. Toki haarautuva juoni ei keksintönä ole uusi. Esimerkiksi *Indiana Jones and the Fate of Atlantis* -seikkailupeli vuodelta 1992 haarautuu kolmeen erilaiseen polkuun [31]. Myös Daedalic julkaisee pelin, joka keskittyy interaktiiviseen tarinankerrontaan ja jossa on haarautuva juoni: *Silence: The Whispered World 2* [32]. Kuitenkin myös vanhoista seikkailupeleistä on julkaistu *remake*- tai *remastered*-versioita, joissa niiden käyttöliittymää tai grafiikoita on paranneltu.

Japanissa suosittuja ovat visual novel -pelit. Niissä esitetään tarina tekstinä ja sitä elävöitetään taustakuvilla ja animaatioilla sekä musiikilla ja äänitehosteilla. Niissä ei kirjoiteta komentoja niin kuin tekstiseikkailupeleissä, vaan pelaajalle esitetään lista

vaihtoehtoja, mitä tehdä seuraavaksi. Tarina voi edetä erilaisiin suuntiin valintojen mukaan. Pelikokemukseen keskittyvät visual novel -pelit saattavat alkaa kasvattaa suosiotaan myös länsimaissa [33]. [34] [35]

3. TYÖKALUN OMINAISUUKSIEN MÄÄRITTELY

Luvussa kuvataan seikkailupeleille ja niiden suunnittelulle tyypillisiä ominaisuuksia, joiden pohjalta työkalu suunniteltiin. Luku vastaa ensimmäiseen tutkimuskysymykseen.

3.1 Käyttäjät

Työkalua käyttää ensisijaisesti seikkailupelin suunnittelija tai suunnittelijat suunnitelman laatimiseen ja dokumentoimiseen. Muu kehitystiimi käyttää työkalua suunnitelman tutkimiseen.

Suunnittelija suunnittelee pelin tarinan ja juonen, sen maailman ja jokaisen ruudun, hahmot ja tavarat sekä tehtävät. Hän myös dokumentoi suunnitelman kehitysvaihetta varten. Ideaalinen suunnittelija osaa kirjoittaa niin teknisesti kuin kaunokirjallisesti ja hänellä on hyvä yleistieto. Hänellä täytyy olla mielikuvitusta, jotta hän pystyy keksimään pelin maailman ja säännöt. Hänellä on teknistä tietämystä, jotta hän ymmärtää teknologian mahdollisuudet ja rajoitteet. Analyyttisyys auttaa arvioimaan pelisuunnitelmaa. Hyväksi suunnittelijaksi ei synnytä, vaan taitoja voidaan opiskella ja harjoitella. [14, s. 18–27]

Seikkailupelien kehitystiimissä on samoja vastuualueita kuin muidenkin pelien kohdalla, muun muassa suunnittelu, ohjelmointi, grafiikat, ääni, projektinhallinta ja testaus [36]. Sama henkilö voi hoitaa useampaa roolia [29]. Joissain projekteissa on suunnittelijan lisäksi käsikirjoittaja, joka kirjoittaa tarinaa ja dialogeja [14, s. 24].

Luonnollisesti on erikokoisia produktioita ja siis myös erikokoisia tiimejä. Tulevaa *Silence: The Whispered World 2* -peliä kehittää noin 20 henkilöä [37]. Tiimissä on esimerkiksi pelin ohjaaja (game director), joka päättää pelin suuntaviivat sekä pääsuunnittelija (lead designer), joka ohjaa pelin suunnittelua [37]. Pienempänä esimerkkinä Wadjet Eye Games -pelistudiolla kehitystiimeissä on yleensä kolme tai neljä henkilöä [29]. Ne koostuvat kirjoittaja-suunnittelijasta, ohjelmoijasta sekä graafikosta ja säveltäjästä [29]. Usein kirjoittaja-suunnittelija myös ohjelmoi pelin [29]. Välillä he käyttävät kahta graafikkoa: yksi tekee taustat ja toinen animoi hahmot [29].

3.2 Seikkailupelin yleiskuvaus

On olemassa hyvin erityyppisiä pelejä ja ne jaotellaan yleensä genreihin ominaispiirteiden mukaan. Schell määrittelee pelin ongelmanratkonta-aktiviteetiksi, jota lähestytään leikkisällä asenteella [38, s. 37]. Peli koostuu mekaniikoista, tarinasta, estetii-

kasta ja teknologiasta, joiden on tuettava toisiaan [38, s. 41–42]. Pelimekaniikat ovat pelin säännöt ja menetelmät, millä tavoin pelaaja voi pyrkiä kohti pelin tavoitetta [38, s. 41]. Estetiikka tarkoittaa pelin kuvia, ääniä ja muita elementtejä, jotka esittävät pelin maailman [38, s. 42]. Teknologia on media, joka mahdollistaa estetiikan esittämisen, pelimekaniikkojen suorituksen sekä tarinan kerronnan ja etenemisen [38, s. 42–43].

Useammankin genren pelit voivat sisältää seikkailuja, mutta ne eivät silti ole seikkailupelejä. Genren nimi tuleekin ensimmäisestä seikkailupelistä *Adventure*. Seikkailupeli on interaktiivinen tarina pelaajahahmosta, jota pelaaja kontrolloi. Maailman tutkiminen sekä tehtävien ratkonta keräämällä ja käyttämällä tavaroita ja keskustelemalla hahmoille ovat seikkailupeleille tyypillisiä ominaisuuksia. Seikkailupeleissä ei ole ollenkaan tai on hyvin vähän taistelua ja muita toimintaelementtejä. Erikseen on genre toimintaseikkailu. Toimintaseikkailuissa on tehtäviä ratkaistavana, mutta ne ovat nopeatempoisempia kuin seikkailupelit ja tarjoavat myös fyysisiä haasteita. [14]

Seikkailupeleissä ei ole yleensä kuin yksi pelimoodi, ellei tavaraluettelon tai kartan katsomista lasketa sellaiseksi [14]. Joissain peleissä on päiväkirja, johon tallentuu automaattisesti muistiinpanoja tapahtuneista asioista, joita pelaaja voi hyödyntää tehtävien ratkaisemisessa [14]. Seikkailupelit ovat useimmiten yksinpelejä.

Työkalussa suunniteltava peli voi olla PC-, konsoli- tai mobiilipeli. Kaikki alustat ovat mahdollisia, sillä työkalussa ei määritellä välineitä, joilla pelaaja kommunikoi pelin kanssa tai saa palautetta. Siinä ei näy peliin valittavat teknologiat.

3.3 Tarinan ja juonen suunnittelu

Wadjet Eye Games -pelistudion seikkailupelisuunnittelija Dave Gilbert kertoo suunnittelevansa pelejä muistikirjan ja kynän kanssa. Hän istuu kahvilassa ja kirjoittaa mieleen tulevia ideoita, joista vain muutamat päätyvät peliksi asti. Kun idea on muotoutunut pidemmälle, Gilbert määrittelee pelin tarinan ja juonen, millaisia hahmot ovat ja niin edelleen. Hän ei käytä mitään erityisiä menetelmiä tai työkaluja. [29]

Seikkailupelille tyypillistä on, että siinä on tarina, joka tapahtuu tietynlaisessa maailmassa [14]. Osana tarinaa pelaajalla on isompi tavoite, jota kohti pelaaja pääsee ratkaisemalla pienempiä tehtäviä [14]. Suuri, lähes koko pelin kattava tavoite muodostaa pääjuonen [14, s. 458]. Kun pelaaja ratkaisee pieniä tehtäviä, hän saa uusia tilalle. Nämä vievät juonta eteenpäin [14]. Alun tilanne ja päätehtävä esitetään usein välianimaationa (cut scene) pelin alussa [14]. Välianimaatio tarkoittaa animaatiota, joka ei ole interaktiivinen, vaan pelaaja vain katsoo sitä. Esimerkiksi *The Curse of Monkey Island* -pelissä päätavoitteena on aivan pelin alkua lukuun ottamatta pelastaa pelaajahahmon rakastama Elaine [39] [40]. Peli jakaantuu kuuteen lukuun, joissa on koko luvun kattavat tavoitteet [39] [40]. Ne on esitetty alla olevassa listauksessa.

Ensimmäisen luvun jälkeen jokaisen luvun tavoite vie kohti tavoitetta pelastaa Elaine [39] [40].

> Luku 1: Pakene merirosvokapteeni LeChuckin laivasta.
>
> Luku 2: Hanki laiva, miehistö ja kartta saarelle, jolla on kirouksen poistamiseen tarvittava timantti, sekä kirottu Elaine-patsas takaisin.
>
> Luku 3: Kartta varastetaan, hanki se takaisin, jotta pääset saarelle.
>
> Luku 4: Etsi haaksirikossa kadonnut Elaine-patsas ja poista siitä kirous.
>
> Luku 5: LeChuck langettaa päähahmolle kirouksen, joka pitää purkaa.
>
> Luku 6: Voita LeChuck.

Ron Gilbert kertoo suunnittelevansa seikkailupelin luvuissa (act) [41]. Kun pelin jakaa lukuihin, pelaaja kokee saavuttavansa jotain, jo kun hän saa yhden luvun päätökseen [41]. Pelaajan ei tarvitse muistaa liikaa asioita, joita on kerrottu aiemmin pelissä [41]. Lisäksi tavaraluetteloa voidaan tarpeen tullen tyhjentää, jotta kaikki tavarat eivät pysy mukana loppuun asti, vaikka niille ei enää olisi käyttöä [41]. Myös *Grim Fandango* -pelin suunnitteludokumentissa peli on jaettu tapahtumavuosiin, jotka ovat käytännössä lukuja [42].

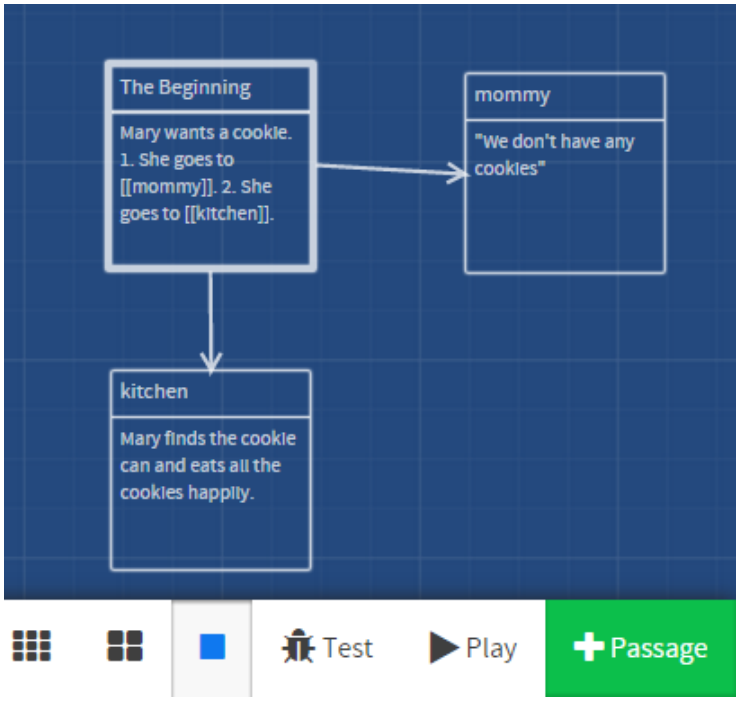
Työkalussa tarina jaetaan lukuihin. Käyttäjä luo lukuja ja kirjoittaa jokaiseen lukuun kertomuksen luvun tapahtumista. Luvuista muodostuu pelin koko tarina.

Interaktiivisten tarinoiden luontiin on saatavilla Twine, jota voi käyttää myös verkossa. Siinä kirjoitetaan jae kerrallaan, ja tekstiin voidaan lisätä linkki toiseen jakeeseen tietyllä syntaksilla. Twine näyttää jakeet ja niiden yhteydet graafina. Kuvassa 3.1 on esimerkki Twinen graafista. [43]

Koska seikkailupelien oleelliset osat ovat tarina ja tehtävien ratkaisu, niiden uudelleenpelattavuus ei yleensä ole mielekästä [14, s. 446]. Lisää pelattavuutta peliin tuovat haarautuvat tarinaketjut.

Työkalussa lukujen yhteydet esitetään graafina. Jokaista lukua kuvaa lukuelementti, joka voidaan liittää toisiin lukuelementteihin. Pelin juoni voi haarautua, jolloin luvulla on useampi loppuvaihtoehto. Toisaalta haarautunut juoni voi palata samaan yhteen lukuun, eli luvulla voi olla myös useampi alku. Kukin loppuvaihtoehto liitetään sen luvun alkuun, johon tarina kyseisestä lopetuksesta etenee.

Snowflake on menetelmä tarinan suunnitteluun. Snowflake-menetelmässä aloitetaan kirjoittamalla yksi lause, joka kuvaa koko tarinan [44]. Sen jälkeen kasvatetaan lausetta ja kirjoitetaan yksi kappale, joka kuvaa tarinan asetelman, tärkeimmät käännekohdat ja



Kuva 3.1: Twine-työkalu, jolla voidaan luoda interaktiivisia tarinoita. Kuvakaappaus Twine-työkalun online-versiosta.

lopun [44]. Tämän jälkeen kirjoitetaan henkilöhahmot, ja näin edelleen jatketaan tarinan kehittämistä kasvattamalla jo olemassa olevaa tekstiä [44]. Työkalussa suunnitelma alkaa lukujen otsikoista ja kuvauksista, jotka kuvaavat tarinan pääkohdat. Jokaiseen lukuun määritellään tehtäviä, joiden kautta lopulta muodostuu koko pelin juoni.

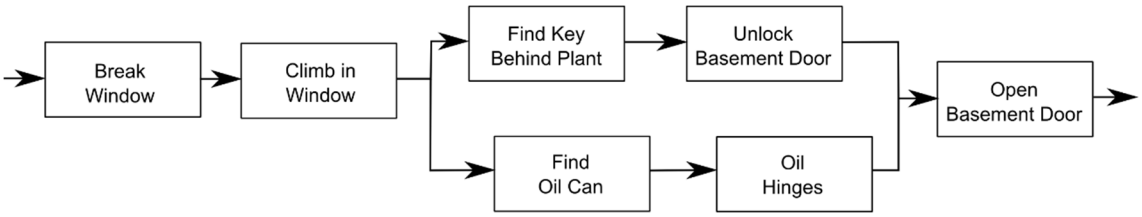
3.4 Tehtävien suunnittelu

Seuraavassa on esitetty tehtävien suunnitteluun liittyviä ominaisuuksia. Tehtävä kuvataan ja eri tehtävien välille määritellään riippuvuuksia. Tehtävää suorittaa tietty pelaajahahmo. Tehtävän suorittamiseksi pelaaja käyttää komentoja ja tavaroita.

3.4.1 Tehtävien riippuvuudet

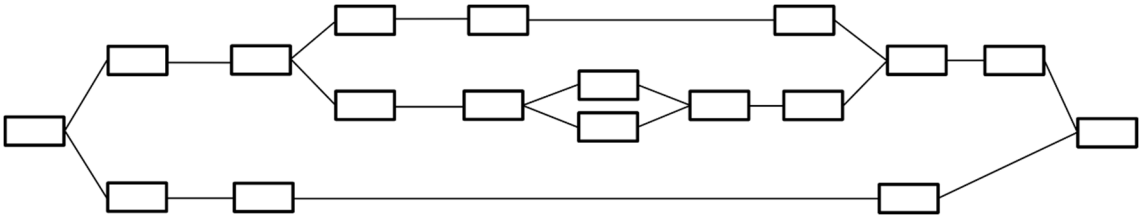
Ron Gilbert esittää Grumpy Gamer -blogissaan menetelmiä, joilla hän on suunnitellut seikkailupelejä. Gilbert tunnetaan LucasArtsille suunnittelemistaan peleistä, kuten *Monkey Island* -sarja ja *Maniac Mansion*. Tehtävien suhteiden suunnitteluun voidaan käyttää *Puzzle Dependency Chart* -graafia. Se auttaa estämään umpikujia muodostumasta peliin sekä tasapainottaa tehtävien kulkua, esimerkiksi ettei tehtävägraafiin synny pullonkaulaa. Pullonkaulan kohdalla kaikki rinnakkaiset tehtävät päätyvät yhteen samaan tehtävään. Gilbert aloittaa graafin piirtämisen ja siis tehtävän suunnittelun tehtävän loppupäästä. On tavoite, ja sitä ennen täytyy tehdä askelia, jotta tehtävän saa ratkaistua. Kuvassa 3.2 on esimerkki Puzzle Dependency Chart -graafista. Siinä

tehtävän suunnittelu on aloitettu tehtävästä ”Open Basement Door”. Sen jälkeen on mietitty, mitä oven avaamiseen tarvitsisi ja tehty niille taas riippuvaisia tehtäviä.



Kuva 3.2: Tehtäväkokonaisuus Puzzle Dependency Chartissa. Nuoliviivat kuvaavat tehtävien riippuvuuksia. Perustuu lähteeseen [41].

Graafi auttaa myös muodostamaan rinnakkaisia tehtäväpolkuja. Kuva 3.3 esittää pelkistettynä kokonaisen graafin. Yhden tehtävän ratkaisusta tulisi aueta kahdesta kolmeen uutta tehtävää, jotka taas palaavat yhteen tehtävään. Tällöin muodostuu salmiakin muotoisia rakenteita sisäkkäin, kuten kuvassa.



Kuva 3.3: Pelkistetty kuva kokonaisesta Puzzle Dependency Chartista. Tehtävät haarautuvat ja palaavat taas yhteen, jolloin muodostuu salmiakin muotoisia tehtäväriippuvuuksia sisäkkäin. Perustuu lähteeseen [41].

Ron Gilbert sanoo, että aikoinaan Lucasfilm Games -studiolla graafien piirtämiseen käytettiin jotakin ohjelmistoa, joka oli tarkoitettu projektien aikatauluttamiseen. Nykyisin hän käyttää graafien tekemiseen OmniGraffle-ohjelmaa, joka käyttää Graphviziä. OmniGraffle toimii vain Macilla. OmniGrafflella voidaan piirtää kaavioita erimuotoisilla ja värisillä objekteilla, joihin voidaan kirjoittaa tekstiä, ja se tukee erilaisia automaattisia asetteluja (layout) [45]. Vaihtoehtoisesti Gilbert syöttää graafin tiedot suoraan tekstinä Graphviz-työkaluun, joka generoi graafin. Tämä vaatii kuitenkin enemmän teknistä osaamista. [41]

Tehtävän ratkaisu on pelaajalle kiinnostavampi ja hauskempi, jos siinä on monta tasoa, kuin jos ratkaisu olisi hyvin yksinkertainen. Tällöin ennen varsinaista ratkaisua on rinnakkain ja peräkkäin tehtäviä, jotka ovat osa ratkaisua. Varsinainen tehtävä on siis riippuvainen näistä osatehtävistä. Esimerkki yksinkertaisesta tehtävästä voisi olla, että tulen sammuttamiseksi tarvitsisi mennä lähellä olevalle kaivolle ja ottaa siellä olevalla ämpärillä vettä. Sen sijaan kaivolla ei olisikaan ämpäriä eikä köyttä ja kampeakaan, vaan ämpäri pitäisi kaiverruttaa halosta, mitä varten tulee hankkia halko ja viedä se puusepälle, joka haluaa jonkun tavaran, jotta suostuu tekemään palveluksen ja niin edelleen. Kun tehtävät eivät ole enää niin ilmiselviä, suunnittelija voi antaa pelaajalle

esimerkiksi dialogeissa tai pelaajahahmon kommenteissa epäsuoria vinkkejä, miten asiat toimivat ja miten niitä voidaan käyttää. Se tekee tehtävistä järkeviä ja auttaa pelaajaa ymmärtämään ratkaisun logiikan. [46]

Työkalussa tehtävät esitetään Puzzle Dependency Chart -kaaviota vastaavassa tehtävägraafissa. Tehtävägraafi muodostetaan luomalla tehtäviä ja määrittämällä niille riippuvuuksia. Jokaisella luvulla on yksi tehtävägraafi. Luvulle määritellyt alut ja loput näkyvät tehtävägraafissa valmiina. Tehtävägraafin tehtävässä on lyhyt kuvaus tehtävän tavoitteesta. Tehtäviä voidaan siirrellä, jotta tehtävägraafi voidaan järjestellä selkeään muotoon. Tehtävägraafissa voidaan luoda peräkkäisiä ja rinnakkaisia tehtäviä. Rinnakkaisia tehtäviä luodaan määrittämällä tehtävälle riippuvaisuus useaan tehtävään. Kun tehtävään tulee monta nuolta, se on riippuvainen kaikista kyseisistä tehtävistä. Tehtävät eivät ole vaihtoehtoisia, jos ne ovat lopulta riippuvaisia samasta tehtävästä. Tällöin ne päätyvät samaan luvun lopetukseen. Jos tehtävät haarautuvat ja päätyvät eri lopetuksiin, ne ovat vaihtoehtoisia keskenään.

Irralliset tehtävät, jotka eivät tunnu liittyvän mitenkään pelin tarinaan, eivät ole mielekkäitä pelaajalle [47]. Esimerkiksi miksi oven vieressä oleva shakkitehtävä avaisi oven [47]? Suunnittelija voi kuitenkin keksiä tavan, joka selittää tämän ja kertoa sen pelaajalle esimerkiksi dialogissa jonkun hahmon kanssa, jolloin tehtävä on taas järkevä [47]. Tehtävä saattaa myös vaatia liian pitkälle vietyä päättelyä, jolloin ratkaisu ei ole pelaajalle enää järkevä [14, s. 472–473]. Jotkut tehtävät vaativat triviatietoa pelin ulkopuolelta [14, s. 473–474]. Tämä tieto voi olla sidottu kulttuuriin tai aikaan, eivätkä toisten kulttuurien edustajat tai peliä myöhemmin pelaavat voi keksiä tehtävän ratkaisua [14, s. 473–474]. Työkalu ei pysty arvioimaan, onko tehtävä mielekäs ja looginen. Tähän vaadittaisiin erittäin kehittynyt tekoäly. Tehtävien mielekkyyden arviointi on täysin suunnittelijan vastuulla.

3.4.2 Tehtävä

Grim Fandango -pelin suunnitteludokumentissa tehtävät on tehtävägraafissa kuvattu yleisellä tasolla. Sen jälkeen jokainen tehtävä on kuvattu yksityiskohtaisesti tekstinä. Kuvassa 3.4 on esitetty yksi tehtävä dokumentista. Tehtävällä on numero ja nimi. Tehtävä on jaettu kahteen osaan: tilanteen ja ongelman kuvaukseen sekä ratkaisuun. Tehtävien kuvauksessa kerrotaan samalla myös kuinka juoni etenee. Kuvaus ratkaisu mukaan lukien on tarinan muodossa. Tehtävien kuvauksien välissä on kuvattu myös välianimaatiot tekstinä. [42]

Tehtävä luodaan työkalussa aina ensin tehtävägraafiin. Tehtävägraafista valitaan tehtävä, jota halutaan muokata. Tehtävällä on nimi, eli lyhyt kuvaus, joka esitetään tehtävägraafissa. Tehtävä on jaettu ongelma- ja ratkaisukenttiin *Grim Fandango* -suunnitteludokumenttia vastaavasti. Ne ovat vapaita tekstikenttiä, jotka eivät rajoita suunnittelijan ilmaisua ja kerrontaa.

Puzzle #5: Jam Door Open

"WHAT A FRIGGIN MESS!" Juan shouts, "IT'S THOSE DAMN MAILROOM PUNKS AGAIN!" He says this no matter how many times Manny pulls this stunt. But he won't let Manny mess with anything while he's in there, and when he leaves, he always closes the door, and the doorknob lock is still locked.

Solution:

While Brennis is working on the server, Manny uses the key in the deadbolt and extends the bolt with door open. Then Brennis walks off muttering loudly about mail-room punks, and doesn't notice that the extended deadbolt stops the door from closing completely and locking. Enter the Reaper…

Kuva 3.4: Tehtävä Grim Fandangon suunnitteludokumentista. Tehtävällä on numero ja nimi. Tehtävän tilanne ja ongelma on kuvattu, ja sen jälkeen ratkaisu. [42]

Juoni etenee, kun pelaaja ratkaisee tehtäviä. Tehtävien ratkaisu vaatii usein lateraalista ajattelua, eli epäsuoraa ja luovaa päättelykykyä, toisin kuin vaikka tasohyppelypeli, joka voi vaatia tarkkuutta, nopeutta ja hyvää näppäimistönkäyttökykyä [14]. Jotta pelaaja voi päätellä ratkaisuja tehtäviin, tehtävien ongelmat ja ratkaisut perustuvat oikean maailman toimintaan, tai pelimaailman sääntöihin, jotka tulevat pelaajalle pelissä ilmi [48, s. 275]. Ratkaisut voivat kuitenkin olla yllättäviä [48, s. 275]. Esimerkiksi pelaajan on luotava ritsa, mikä onnistuu, kun hän tajuaa oikeasta maailmasta oppimansa tiedon, että tyhjentynyt sisäkumi on venyvää materiaalia [48, s. 275]. Tehtävät voidaan jakaa tavara-, pulmapeli- ja dialogitehtäviin. Lisäksi pelissä voi olla esimerkiksi labyrintti-tehtävä, jossa pelaajan on löydettävä reitti haluttuun paikkaan sokkelomaisella alueella, mutta tällaisesta tehtävästä on vaikea tehdä mielenkiintoista ja mielekästä [14].

Tavaratehtävissä pelaaja kerää tavaroita maailmasta ja hahmoilta, yhdistelee tavaroita toisiinsa ja käyttää niitä maailmassa oleviin kohteisiin tai antaa hahmoille saavut-taakseen tehtävän tavoitteen. Tavarat ja niiden käyttökohteet ovat tyypillisesti eri ruuduissa eli pelaaja joutuu liikkumaan maailmassa ja etsimään kohteet ja tavarat. Pelaajalla on tavaraluettelo, jossa näkyy pelaajan keräämät tavarat. Luettelon tavaroita voidaan yhdistellä tavaraluettelon sisällä, tai käyttää maailmassa oleviin asioihin. Tavaratehtävä voi olla perinteinen "Löydä avain lukittuun oveen", jossa ovi tarkoittaa mitä tahansa estettä pelaajan tavoitteen tiellä ja avain mitä tahansa tavaraa, joka poistaa esteen [14].

Työkalussa tavaratehtävä kirjoitetaan tehtävän ongelma- ja ratkaisukenttiin. Tehtävässä käytettävät tavarat luodaan samalla, kun tehtävää kirjoitetaan. Tehtävästä näkyy selkeästi, mitä tavaroita siinä käytetään. Pelaaja voi saada tehtävässä tavaroita tavaraluetteloonsa tai menettää niitä.

Pulmapeleissä pelaaja ratkaisee usein yhdessä ruudussa olevan älypelin. Pulmapeli ei välttämättä liity millään tapaa pelin juoneen, mikä voi heikentää immersiota ja motivaatiota ratkaista tehtävä. Pulmapeli on vaikea tehdä luontevaksi osaksi pelin juonta ja maailmaa. Esimerkiksi *Resonance*-pelissä on pulmapeli, joka on luonteva osa maailmaa. Siinä on tehtävänä avata koodilukolla ovi. Koodilukon numerotaulu on kuitenkin rikki. Numerotaulu tulee irrottaa ruuvimeisselillä, jolloin pulmapeli tulee esiin. Oven avaamiseksi koodilukon piirilevyn pinnit tulee liittää johdolla toisiinsa tietyllä tavalla. Ratkaisu vaatii hoksaamista ja yksinkertaista matematiikan osaamista, mutta ei erityistä tietämystä piirilevyistä. Pulmapeli on esitetty kuvassa 3.5. [49] [50]

Kuva 3.5: Kuvakaappaus Wadjet Eye Gamesin julkaisemasta pelistä Resonance. Pulmapeli, jossa pitää yhdistää koodilukon johto oikein, jotta ovi aukeaa. Kuva otettu lähteestä [51].

Työkalussa on keskitytty ensisijaisesti tavaratehtävätyypin tukemiseen, koska ne ovat yleisempiä kuin pulmapelit. Kuvat auttaisivat pulmapelin havainnollistamiseen, mutta tehtäviin ei pystytä liittämään kuvia. Pulmapeli voidaan kuitenkin esittää tekstinä ongelma- ja ratkaisukentissä. Tekstissä voidaan viitata työkalun ulkopuolisiin resursseihin.

Monissa seikkailupeleissä on NPC-hahmoja (non-player-character), eli hahmoja, joita pelaaja ei ohjaa. Hahmot elävöittävät maailmaa, kertovat tarinaa ja ovat osana tehtäviä [14, s. 469]. Usein pelaajahahmo voi jutella niiden kanssa ja niiltä saa informaatiota

maailmasta, muista hahmoista ja tehtävistä. Dialogitehtävissä keskustellaan pelissä olevien hahmojen kanssa, jotta saavutettaisiin tavoite. Dialogitehtävä voi tapahtua yhden keskustelun aikana tai pelaajahahmon täytyy käydä juttelemassa eri hahmoille sopivalla tavalla tehtävän ratkaisemiseksi. Hahmoilta voidaan myös saada tavaroita tai ne voivat haluta jonkun tavaran, jolloin ne voivat esimerkiksi päästää pelaajan uuteen paikkaan tai pelaaja voi saada toisen tavaran. Ne voivat olla oleellinen osa juonta.

Pelaaja odottaa NPC-hahmojen olevan humaaneja eikä liian konemaisia. Hyvän dialogitekoälyn teko on hankalaa, joten dialogit on yleensä kirjoitettu valmiiksi puumaiseen rakenteeseen. Pelaaja voi valita vaihtoehdoista, mitä haluaa sanoa. NPC vastaa sen mukaisesti ja dialogi voi sitten haarautua esimerkiksi jatkokysymyksiin kysytystä aiheesta. Valmiiksi kirjoitetuissa dialogeissa on se etu, että keskustelu on luontevaa ja pelisuunnittelija voi ilmentää hahmojen persoonallisuuksia keskusteluissa. [14, s. 469]

Työkalussa ei ole erillistä dialogin muokkausmahdollisuutta. Tehtävän ongelma- ja ratkaisukenttiin voidaan kirjoittaa vapaamuotoinen dialogi.

Seikkailupeli voi koostua vain jostakin tehtävätyypistä, tai olla sekoitus niitä. Esimerkiksi *Machinarium* koostuu pitkälti aina yhdessä ruudussa suoritettavista pulmapelin tyylisistä tehtävistä, mutta niissä käytetään kuitenkin ruudussa esiintyviä tavaroita.

3.4.3 Pelaajahahmo

Pelaajahahmoja on yleensä yksi, mutta on myös pelejä, joissa on useampia pelaajahahmoja. Esimerkiksi *Day of The Tenctacle*- ja *Resonance*-peleissä voidaan valita, mitä hahmoa pelaaja haluaa ohjata sillä hetkellä. Eri hahmojen voi täytyä suorittaa tehtäviä vuorotellen eri paikoissa, jotta päästään eteenpäin. Useampien hahmojen ansiosta voidaan siis suunnitella erilaisia, ehkä pelaajalle mielenkiintoisella tavalla monimutkaisempia tehtäviä kuin jos hahmoja olisi vain yksi. Pelaajahahmolla voi myös olla apuri, jota ei suoranaisesti ohjailla, mutta joka voidaan laittaa tekemään erilaisia asioita. Esimerkiksi *The Whispered World* -pelissä päähahmolla on kaverina perhosentoukka, joka voi ottaa erilaisia muotoja, ja toimii näin ratkaisuna useampaankin tehtävään. Pelaajahahmolla on yleensä persoonallisuus ja mahdollisesti rooli, kuten tietty ammatti, ja ne ovat osa tarinaa [14, s. 448, 455]. Kuitenkin ensimmäisen persoonan peleissä hahmolla ei välttämättä ole keksittyä persoonaa vaan hahmo on pelaaja itse [14, s. 451].

Työkaluun ei erikseen määritellä pelaajahahmoa tai hahmoja. Ne esitellään, kun tarinaa kirjoitetaan lukuihin ja tehtäviin. Tehtävän kuvaukseen tai ongelma- ja ratkaisukenttiin voidaan kirjoittaa, mikä pelaajahahmo suorittaa tehtävän, jos pelaajahahmoja on useampi. Tehtävien riippuvuudet voidaan esittää samalla tavalla riippumatta siitä, monta pelaajahahmoa pelissä on.

3.4.4 Komennot

Nykyään seikkailupelit ovat useimmiten point-and-click-tyylisiä pelejä, eli niissä vuorovaikutetaan maailman kanssa klikkailemalla ruudulla olevia asioita [14]. Pelaaja valitsee suoritettavan toiminnon komentojen valikosta. Valittavana on esimerkiksi silmän, suun ja käden ikonit, jotka vastaavat katsomista, puhumista tai syömistä sekä ottamista tai käyttämistä. Työkalussa ei määritellä valittavana olevia komentoja eksplisiittisesti. Käyttäjä kirjoittaa tehtävät vapaisiin tekstikenttiin, joten hän voi käyttää mitä tahansa sanoja toimintojen kuvaamiseen.

3.4.5 Tavarat

Scrivener-ohjelma on sisällönluontityökalu, joka on tarkoitettu kirjailijoille, käsikirjoittajille, tutkijoille ja muille henkilöille, jotka käsittelevät pitkiä ja monimutkaisia dokumentteja. Työkalua voidaan käyttää siis myös seikkailupelien suunnitteluun ja dokumentointiin. Työkalu auttaa löytämään ja järjestelemään sisältöä, ja siinä voidaan käyttää valmiita tai itsetehtyjä pohjia tietynmuotoisen sisällön luomisen pohjana. Pohjana voi olla esimerkiksi hahmokortti, johon kirjoitetaan hahmon nimi, ominaisuudet ja lisätään kuva. [52] [53] [54]

Työkalussa tavaralle on määritelty pohja, johon lisätään tavaraan liittyviä tietoja. Tavaralla on nimi, kuva, kuvaus ja ominaisuuksia. Samantyylinen rakenne on myös huoneella, josta on kerrottu kohdassa 3.5.

Tavaroiden ja tehtävien suunnittelussa suunnittelijan kannattaa ottaa huomioon tavaroiden uudelleenkäytettävyys eri tehtävissä. Uudelleenkäytettävyys on pelaajalle hauskaa ja mielekästä [46]. Jokainen tavara ei ole olemassa vain yhtä tehtävää varten, vaan niitä esimerkiksi yhdistellään toisiinsa uusien tavaroiden luomiseksi tai käytetään eri tavalla toisessa tehtävässä, tai vaikka samalla tavalla. On mielekkäämpää käyttää aiemmassa tehtävässä käytettyä kuppia veden kantamiseen kuin pakottaa pelaaja hankkimaan toinen tavara veden kantamista varten, etenkin jos jo käytetty tavara on vielä pelaajan tavaraluettelossa.

Työkalun tehtävässä esitetään tavarat, jotka pelaaja on saanut tehtävää edeltävissä tehtävissä. Näin suunnittelija näkee käytössä olevat tavarat ja voi pohtia niiden uudelleenkäyttöä. Tavaralle voidaan lisätä ominaisuuksia. Ominaisuus kuvaa, millainen tavara on. Esimerkiksi kupin ominaisuus voisi olla ”can carry liquid”. Tehtävän ongelma- ja ratkaisukentissä voidaan hakea tavaroita ominaisuudella. Kun suunnittelija tarvitsee tehtävään tietynlaista tavaraa, hän voi tehtävää kirjoittaessaan hakea, mitä sellaisia tavaroita on olemassa.

Pelimaailmassa on tavaroita, joita pelaaja ei voi käyttää tai ottaa mukaansa; ne ovat vain osa taustaa [14]. Monissa peleissä osaa niistä voidaan kuitenkin katsoa, jolloin

pelaajahahmo kommentoi jotain tavaraan liittyvää. Ne vahvistavat immersiota ja voivat kertoa jotain tarinasta. Esimerkiksi yhtyejulisteet hahmon huoneen seinällä voivat kertoa, että hahmo pitää tietynlaisesta musiikista. Pelaajahahmo voi katsoa ja kommentoida julisteita, mikä voi luoda pelaajalle tietyn mielikuvan hahmosta. Näitä kutsutaan työssä *hotspot*-kohteiksi, Adventure Game Studio -pelimoottorissa (AGS) käytetyn termin mukaan.

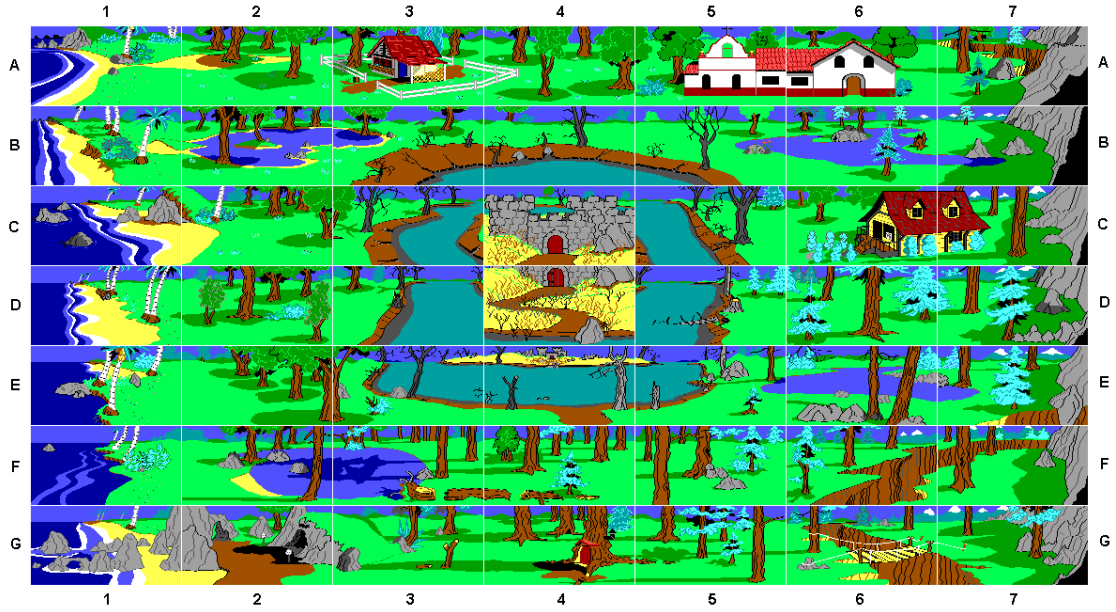
Työkalussa hotspot-kohteita ei määritellä erikseen. Ne voidaan joko kirjoittaa tehtävään pelkkänä tekstinä tai ne voidaan luoda tavaroina. Tehtävässä tavaroille voidaan merkitä, saako pelaaja tavaran. Hotspot-kohteet ovat tavaroita, joita pelaaja ei ikinä saa tavaraluetteloonsa. Hotspot-kohteen kuvaukseen voidaan esimerkiksi kirjoittaa, mitä pelaajahahmo kommentoi, kun hän katsoo sitä.

3.5 Maailman suunnittelu

Maailma, jossa seikkailu tapahtuu, on suuri osa peliä, koska se luo tietyn tunnelman peliin [14]. Esimerkiksi *Gemini Rue* -pelissä on synkeä tunnelma ja *Monkey Island* -sarjan peleissä kepeä ja hauska tunnelma. Tarina voi sijoittua tiettyyn historialliseen ajankohtaan (esimerkiksi *Mata Hari, Indiana Jones*) tai se voi olla tiettyä tyyliä kuten fantasiamaailma (*The Night of the Rabbit, King's Quest* -pelit) tai tieteismaailma (*Primordia, Space Quest* -pelit) [1]. Olivat maailmat realistisia tai eivät, oikeassa elämässä päteviä asioita hyödynnetään yleensä myös seikkailupelien tehtävissä (tuli voidaan sammuttaa vedellä, ovi voidaan avata avaimella). Seikkailupelin maailma on jaettu toisiinsa liittyviin paikkoihin, joiden välillä pelaaja voi kulkea.

Ensimmäisissä seikkailupeleissä pelaaja saattoi liikkua vapaasti kaikkien ruutujen välillä, mutta kun peleistä tuli laajempia ja niihin tuli juoni, ne jaettiin usein lukuihin (act, part tai chapter) [14, s. 456]. Luku koostuu tietyistä ruuduista, joissa pelaaja voi liikkua vapaasti [14, s. 456]. Kun pelaaja on ratkonut luvun tehtävät ja etenee seuraavaan lukuun, hän pääsee uuden luvun ruutuihin, mutta ei enää edellisen luvun ruutuihin [14, s. 456]. Pelaajalle tulee myös tunne, että hän etenee pelissä, ja peli pysyy mielenkiintoisena, kun tutkittavaksi tulee uusia paikkoja. Eri lukujen paikat voivat tosin olla myös samoja. Työkalussa kartta on lukukohtainen.

Maailma voi olla jaettu vierekkäisiin ruutuihin, joiden välillä pääsee liikkumaan kävelyttämällä pelaajahahmo ruudun reunalle [1], kuten kuvassa 3.6. Työkalussa kartta luodaan huoneista. Jokainen ruutu on huone. Huoneiden välille luodaan yhteyksiä merkitsemään, miten huoneiden välillä voidaan liikkua. Vaihtoehtoisesti maailma voidaan jakaa lokaatioihin [1], joihin voidaan matkustaa esimerkiksi valitsemalla erillisessä karttanäkymässä kohde kartalta. Pelin karttanäkymää kutsutaan jatkossa maailmankartaksi. Nämä lokaatiot voivat kuitenkin vielä koostua vierekkäisistä ruuduista. Kuvassa 3.7 on *The Whispered World* -pelin kartta pelin alkupuoliskolta. Pai-



Kuva 3.6: Kuvakaappaukset Sierran pelistä King's Quest II: Romancing the Throne on yhdistetty esittämään pelin karttaa. Ruutujen sivuilta pääsee viereisiin ruutuihin. Maailma rajoittuu vasemmalla mereen ja oikealla vuoriin. [55]

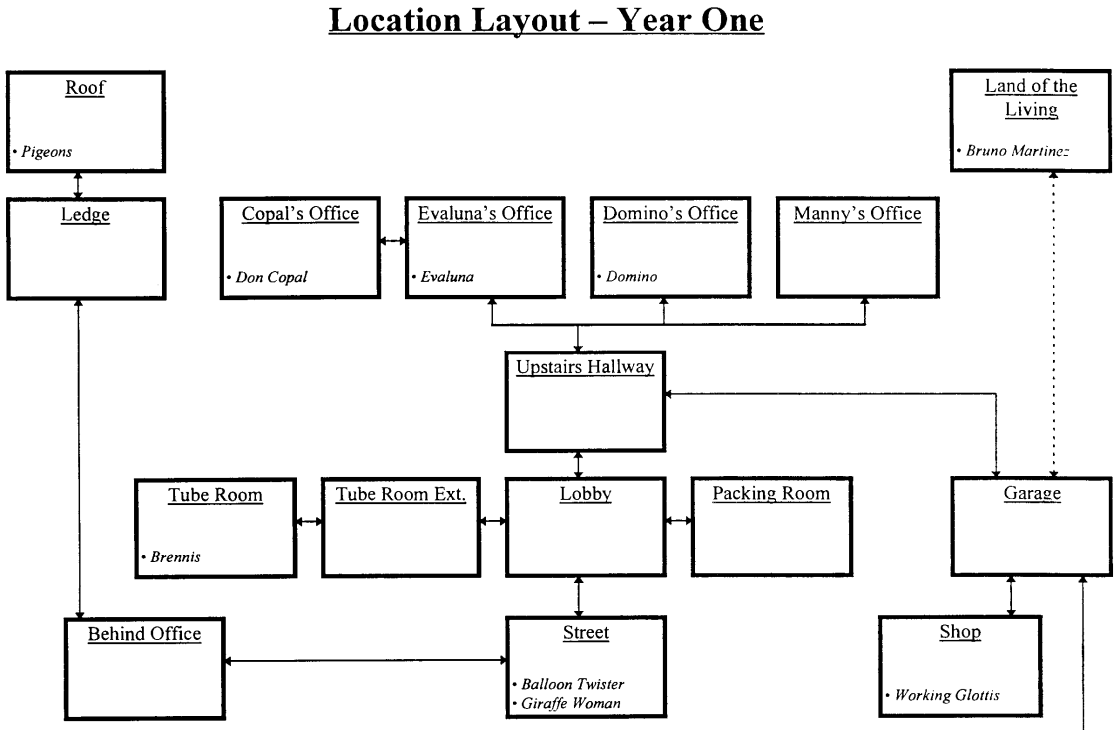
kat on osoitettu ympyröillä. Kun pelaaja vie hiiren kursorin ympyrään, hän näkee paikan nimen. Valitsemalla paikan pelaaja siirtyy sinne.

Työkalu ei tue erillistä maailmankarttaa. Maailmankartan lokaatiot voidaan esittää ruutuina samaan tapaan kuin puhtaasti ruutupohjaisessa kartassa. Maailmankartan lokaatioiden alkuruuduista on yhteydet kaikkiin muihin lokaatioiden alkuruutuihin.

Kuva 3.7: Kuvakaappaus Daedalic Entertainmentin pelistä The Whispered World. Kuvassa näkyy kartta, jossa on ympyröillä merkattu paikat, joihin pelaaja voi mennä. Kuva otettu suoraan pelistä.

Grim Fandango -pelin suunnitteludokumentissa jokaiselle luvulle on esitetty kartta. Kartta kuvaa luvussa esiintyvät paikat nimeltä sekä paikkojen yhteydet toisiinsa. Lisäksi paikkoihin on merkitty hahmoja, jotka esiintyvät kyseisessä paikassa. Kuvassa 3.8 on osa ensimmäisen luvun karttaa. [42]

Työkalun kartta esitetään kuten *Grim Fandango* -pelin suunnitteludokumentissa. Huoneella on nimi ja huoneiden välille luodaan yhteyksiä. Lisäksi huoneelle voidaan liittää sen kuva.



Kuva 3.8: Osa ensimmäisen luvun karttaa Grim Fandangon suunnitteludokumentissa. Huoneilla on nimet ja pääsy toisiin huoneisiin on esitetty nuoliviivoilla. Huoneisiin on merkitty, mitä hahmoja niissä on. [42]

Perinteiset ensisijaisesti tavaratehtäviin perustuvat seikkailupelit on kuvattu kolmannesta persoonasta kontekstisidonnaisesti, eli ruutu näytetään tietystä kulmasta eikä kamera seuraa pelaajaa [14, s. 449]. Maailman tutkimiseen ja yksittäisten ruutujen pulmapelitehtäviin perustuvat pelit voivat usein olla myös ensimmäisestä persoonasta kuvattuja, kuten *Myst* ja *Nancy Drew* -sarja [56]. Kuvakulma ei vaikuta työkalun toimintaan.

Perinteisissä seikkailupeleissä ei liikuta vapaasti 3D-ympäristössä kuten esimerkiksi ensimmäisen persoonan ammuntapeleissä (first person shooter) tai roolipeleissä voidaan liikkua. Jotkut kauhuseikkailu- tai tutkimuspelit tarjoavat vapaan liikkumisen 3D-maailmassa. Vapaaseen 3D-maailmaan sijoittuvalla seikkailupelillä on haasteensa ja haittansa, joiden vuoksi seikkailupelit eivät tyypillisesti ole sellaisia. Esimerkiksi pelistä tulisi helposti toimintapainotteisempi kuin seikkailupeleille on tyypillistä, kun pelaaja

juoksentelisi ympäriinsä kiinnittämättä niin paljoa huomiota ympäristöönsä [14, s. 451]. Työkalu ei tue karttaa, jota ei ole jaettu ruutuihin.

4. YHTEYS PELIMOOTTORIIN

Jos pelisuunnitelma on tehty esimerkiksi paperilla, jokainen huone, tavara, hahmo ynnä muu asia pelissä on luotava yksitellen pelimoottorissa ja syötettävä sille sen tarvitsemat tiedot manuaalisesti. Seikkailupelien suunnittelutyökalun on tarkoitus mahdollistaa suunnitelman helppo vienti pelimoottoriin. Työkalussa tulee olla pelimoottorien pelin objekteja vastaavat objektit. Kaikkia pelimoottorissa esiintyviä objekteja ei määritellä työkalussa lainkaan.

Tässä luvussa selvitetään, mitä pelimoottoreita seikkailupelien tekemiseen käytetään. Näistä kuvataan tarkemmin kaksi pelimoottoria: Visionaire ja Adventure Game Studio ja näiden XML-tallennusmuodot. Näiden tietojen avulla vastataan toiseen tutkimus-kysymykseen.

4.1 Pelimoottorit

Seikkailupelin toteutuksessa käytetään hyödyksi pelimoottoreita. Daedalic Entertainmentin pelejä on tehty Visionaire-pelimoottorilla [57]. Telltale käyttää omaa Telltale Tool -pelimoottoria, joka ei ole julkisesti saatavilla [58]. Wadjet Eye Games käyttää Adventure Game Studio -pelimoottoria [59] [60]. Double Fine käyttää avoimen lähdekoodin Moai-kehystä, jossa on tuki monelle alustalle [61] [62]. Se on tarkoitettu etenkin mobiilipelien kehitykseen [63]. Se ei ole suunnattu vain seikkailupelien tekemiseen, mutta sitä voidaan laajentaa C++-luokilla [63].

Lisäksi on muitakin seikkailupeleille tarkoitettuja pelimoottoreita työkaluineen, kuten Wintermute ja sen avoimen lähdekoodin Lite-versio. Unity-pelimoottorin lisäosana on saatavilla Adventure Creator. Tässä tarkastellaan tarkemmin Visionairea ja AGS:ää, sillä ne ovat ilmaiseksi kokeiltavissa, niillä on tehty menestyneitä seikkailupelejä, ja ne on tarkoitettu juuri seikkailupelien tekemiseen.

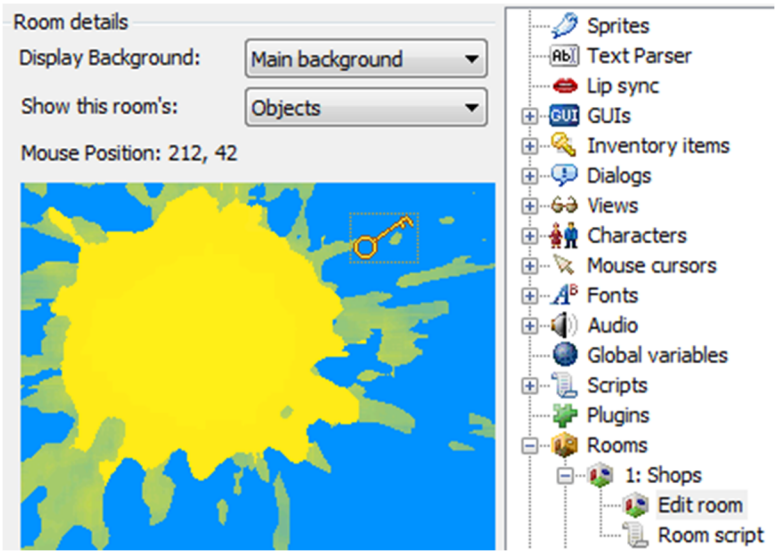
Visionareilla on tehty useita laadukkaita ja suosittuja seikkailupelejä: *Deponia*-sarja, *The Whispered World*, *A New Beginning*, *Harveys New Eyes*, *The Night of the Rabbit* ynnä muita [57]. Visionairea voidaan kokeilla ilmaiseksi, mutta peliä ei voida kääntää suoritettavaksi tiedostoksi, jolla peliä voidaan jakaa [64]. Kaupalliseen käyttöön on tarjolla erilaisia lisenssejä: yhdelle tai useammalle indiekehittäjälle, eli riippumattomalle kehittäjälle, mobiilijakeluun sekä ammattilaiskäyttöön [64].

AGS:llä on tehty menestyneitä pelejä, kuten *Blackwell*-sarja, *Gemini Rue* ja *Resonance* [60]. AGS on ilmainen myös kaupallisessa käytössä, kunhan huomioi sen käyttämien

ulkoisten komponenttien lisenssit [65]. AGS:n korkein resoluutio on 1024x768 ja Visionairen on 1920x1200. AGS:llä tehdään enimmäkseen retrohenkisiä pikseliseikkailuja muun muassa budjettisyistä [66], kun taas Visionairella modernin näköisiä pelejä.

4.2 Adventure Game Studio

AGS:ssä projekti on jaettu eri objekteihin. Ne on esitetty seuraavaksi. Kuvassa 4.1 oikealla näkyy eri objektien valikoita.



Kuva 4.1: Rooms-valikosta on valittu huone muokattavaksi. Huoneeseen on asetettu avain-objekti. Kuvakaappaus Adventure Game Studiosta.

GUIs-valikon alla määritellään käyttöliittymät. AGS:ssä saa halutessaan pohjaksi pelin, jossa on valmiina tyypillisesti tarvittuja asioita. Siinä on valmiina käyttöliittymiä: *Icon bar*:issa on käytettävissä olevat toimintaikonit, kuten kävely, ottaminen ja puhuminen. *Inventory* eli tavaraluettelo näyttää pelaajalla olevat tavarat ja mahdollistaa niiden valitsemisen käyttöön. Pelin tallentamiseen, tallennetun pelin lataamisen ja pelistä poistumiseen on käyttöliittymänäkymät. Lisäksi esimerkiksi vanhoissa peleissä, joissa on laskettu pisteitä, on statusnäkymä sivun ylälaidassa. [67]

Rooms-valikosta lisätään huoneet eli käytännössä kaikki ruudut, joita pelissä näytetään. Huoneet voivat olla pelattavia paikkoja pelin maailmassa, karttoja, välianimaatioiden taustoja tai dialogiruutuja. Kartta on ruutu, jossa on näkyvillä paikat, joihin pelaaja voi mennä. Dialogiruutu näytetään, kun pelaajahahmo keskustelee toisen hahmon kanssa. Huoneelle määritellään taustakuva. Huoneen reunoille voidaan määritellä alueet, joista pääsee kulkemaan toisiin huoneisiin. Huoneeseen määritellään alueet, joilla hahmot voivat kävellä, jotta ne eivät kävelisi esimerkiksi taivaalle. Hahmojen kokoa voidaan skaalata huoneen perspektiivin mukaisesti. Huoneeseen määritellään myös alueet, joiden taakse hahmo voi kävellä. Jos esimerkiksi keskellä huonetta on puu, jonka takaa

hahmo voi kävellä, puu maalataan *Walk-behind*-alueeksi, jotta hahmot eivät piirry sen päälle, kun ne kävelevät sen kohdalta. [67]

Huoneeseen voidaan lisätä objekteja, joita pelaaja voi esimerkiksi ottaa mukaansa tai käyttää. Kuvassa 4.1 huoneeseen on asetettu avain. Objektilla on tunniste, nimi ja kuvaus. Sillä on myös arvo, joka määrittää, onko objekti näkyvissä. Huoneeseen voidaan lisätä hotspot-alueita, joiden kanssa pelaaja voi vuorovaikuttaa. Pelaaja voi esimerkiksi katsoa taivasta, jolloin pelaajahahmo kommentoi säätilaa. Koko taivas voidaan maalata hotspot-alueeksi. Myös esimerkiksi lukko voisi olla hotspot-alue. Sitä ei voida ottaa, mutta pelaajan täytyy käyttää siihen avainta oven avaamiseksi. Hotspot-alue on siis vain osa taustaa, ei erillinen objekti. Hotspot-alueella on tunniste, nimi ja kuvaus. Lisäksi määritellään piste ruudulla, johon pelaajahahmo kävelee, kun pelaaja käyttää hotspot-aluetta. Huoneessa näkyy myös, mitkä hahmot aloittavat sieltä ja mistä kohtaa huonetta. [67]

Inventory items -valikosta voidaan lisätä tavaroita, joita pelaaja voi kerätä maailmasta tavaraluetteloonsa. Tavaralla on nimi, jota voidaan muuttaa. Jokaiselle tavaralle voidaan lisätä kuva, miltä se näyttää tavaraluettelo-käyttöliittymässä, sekä kuva, joka näytetään hiiren kursorin tilalla, kun tavaraa käytetään. Lisäksi tavaralla on boolean-arvo, joka määrittää, onko pelaajalla kyseinen tavara pelin alussa. Tavaralle voidaan lisätä myös itse keksittyjä ominaisuuksia. [67]

Characters-valikosta lisätään hahmoja, mukaan lukien pelaajahahmo tai -hahmot. *Views*-valikossa luodaan animaatioita, kuten hahmon kävely ja puhuminen. Näitä animaatioita liitetään hahmolle *Characters*-näkymän kautta. Hahmolla on tunniste, oikea nimi ja skripteissä käytettävä nimi. [67]

Dialogs-valikosta voidaan lisätä hahmojen välisiä keskusteluja. Yksi dialogi sisältää vaihtoehdot, joita pelaaja voi sanoa, ja skriptin, joka määrittää, mitä tapahtuu eri vaihtoehdoista. Dialogiskriptissä voidaan määritellä, mitä eri hahmot sanovat, jääkö vaihtoehto näkyviin, kun se on sanottu, sekä mihin muihin dialogeihin eri vaihtoehdoista pääsee. Dialogiskriptiin voidaan lisätä myös koodia, jos puheen välissä halutaan toiminnallisuutta, esimerkiksi kauppiashahmo käy hakemassa tavaran kauppansa hyllyltä, tai hahmo vastaa eritavalla, kun joku ehto pätee. [67]

Mouse cursors -valikossa on kursorit eri tilanteisiin. Esimerkiksi kun käyttäjä valitsee aktiiviseksi toiminnoksi puhumisen, kursori muuttuu esimerkiksi suun kuvaksi. Lisäksi on muita valikoita, kuten yleiset asetukset, käännökset ja äänet. [67]

Objekteille, hotspot-alueille ja huoneen reuna-alueille voidaan lisätä tapahtumia (event). Ne määrittävät, mitä tapahtuu, kun asiaa esimerkiksi katsoo tai käyttää. Ohjelmassa 4.1 on esimerkki skriptistä, joka suoritetaan, kun tavaraa ”Key” käytetään. Jokaisella huoneella on oma skripti. Siellä on toteutettu huoneeseen liittyvät tapahtumat. *Scripts*-valikossa on yleisiä skriptejä, jotka eivät liity mihinkään huoneeseen, kuten

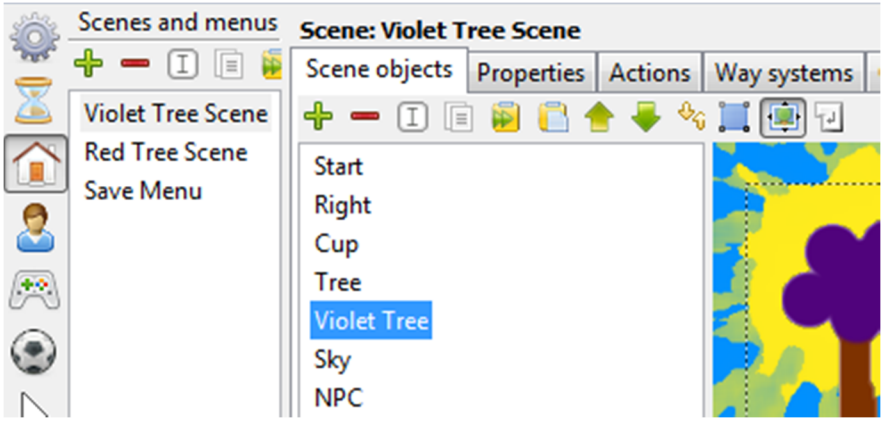
tavaraluettelon näyttäminen ja hahmojen kanssa keskustelu. *Global variables* -valikossa voidaan määritellä erityyppisiä globaaleja muuttujia ja niille alkuarvot. Mikä tahansa skripti voi käyttää niitä. [67]

```
function Key_Interact()
{
    cPlayerGuy.Walk(199,  86, eBlock, eWalkableAreas);
    cPlayerGuy.AddInventory(iKey);
    Key.Visible = false;
}
```

Ohjelma 4.1: Adventure Game Studiossa skripti, joka suoritetaan, kun huoneessa olevaa tavaraa ”Key” käytetään. Pelaajahahmo kävelee tavaran luo, sitä vastaava tavaraluettelon tavara ”iKey” lisätään tavaraluetteloon ja tavara piilotetaan näkyvistä huoneessa.

4.3 Visionaire

Scenes and menus -näkymässä voidaan lisätä paikkoja ja valikkoja (menu). AGS:ssä nämä vastaavat huoneita. Ruudulle valitaan taustakuva. Ruuduille voidaan lisätä toimintoja, jotka suoritetaan, kun ruutuun tullaan tai siitä poistutaan, tai niitä voidaan kutsua muualta. Kuvassa 4.2 on *Scenes and menus* -näkymä ja siitä valitun ruudun näkymä. [68]

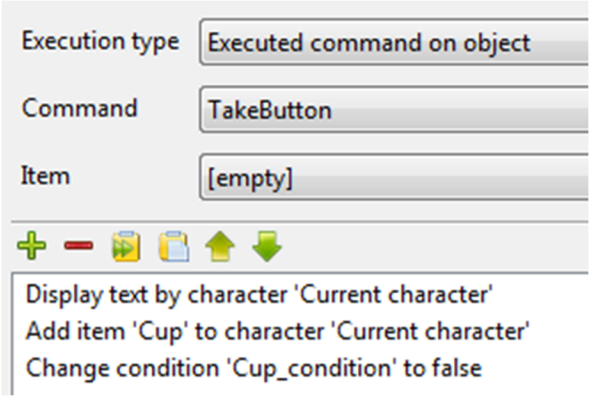


Kuva 4.2: Scenes and menus -näkymässä on listattu pelin ruudut. Ruutuun lisätään objekteja Scene-näkymässä. Kuvakaappaus Visionairesta.

Toiminnot luodaan valmiista vaihtoehdoista. Valikosta valitaan, millainen toiminto on kyseessä. Suoritustyyppi voi olla esimerkiksi hiiren kaksoispainallus. Toiminnolle lisätään osia. Valittavana on muun muassa ehtolauseita, ruudun näyttäminen, animaation pyörittäminen tai pelin tallentaminen. Liittämällä näitä osia sisäkkäin ja peräkkäin muodostetaan toiminto, joka tapahtuu, kun sille valittu suoritustyyppi suoritetaan. [68]

Ruutuun voidaan lisätä objekteja ja asettaa ne halutulle kohdalle taustakuvan päälle. Objektilla on ohjelmassa oleva nimi ja pelissä oleva nimi, sekä kuva. Objektille voidaan lisätä toimintoja. Voidaan esimerkiksi valita, että kun objektiin kohdistetaan käsky

”ota”, näytetään teksti ja lisätään objekti pelaajahahmon tavaraluetteloon. Osa tämäntyyppisen toiminnon määrittämistä on esitetty kuvassa 4.3. Objektille voidaan myös lisätä muuttujia ja ehtoja. Edellisessä esimerkissä, kun pelaajahahmo ottaa objektin, sen boolean-muuttujan arvo voidaan muuttaa epätodeksi. Tämä linkitetään objektin ehtoon, joka määrää, onko objekti aktiivisena ja näkyvissä kyseisessä ruudussa. Objektille voidaan myös lisätä animaatioita ja tehosteita. [68]



Kuva 4.3: Toiminnon lisäämisen osa, jossa määritellään suoritustyyppi ja osat, joista toiminto muodostuu. Kuvakaappaus Visionairesta.

Ruudulle voidaan määritellä alueet, joilla hahmot voivat kävellä, kuten AGS:ssäkin. Lisäksi voidaan lisätä reittipisteitä (way point), joiden muodostamia reittejä pitkin pelaaja kävelee painetulle kohdalle ruutua. [68]

Items-näkymässä voidaan lisätä tavaroita. Tavaralle valitaan kuva ja määritetään nimi pelissä. Myös animaatioita voidaan lisätä. Tavaralle voidaan lisätä toimintoja. [68]

Characters-näkymässä voidaan lisätä ja muokata hahmoja. Hahmoille asetetaan kuvat eri animaatioihin, kuten kävely eri suuntiin ja puhuminen, sekä muita ulkoiseen olemukseen liittyviä asetuksia, kuten kävelynopeus. Hahmoille voidaan lisätä toimintoja ja määritellä muuttujia. [68]

Hahmoille voidaan tehdä dialogeja. Dialogissa voi olla eri tasoja, esimerkiksi pelaajahahmo voi kysyä toiselta hahmolta, saako hän kysyä kysymyksiä, jonka jälkeen mennään alemmalle tasolle, jossa on valittavana eri kysymyksiä. Dialogi muodostetaan osista, jotka koostuvat osan aloitustekstistä sekä vastauksesta. Voidaan merkitä, että osa näytetään vain, jos tietty ehto pätee. Voidaan myös lisätä toiminto, joka suoritetaan, kun dialogi valitaan. Lisäksi määritellään, mitä tapahtuu kyseisen dialogin osan jälkeen, eli pysytäänkö samalla dialogitasolla, palataanko edelliselle tasolle vai poistutaanko koko dialogista, sekä poistetaanko kyseinen vaihtoehto dialogista. [68]

Interfaces-näkymässä lisätään ja muokataan käyttöliittymiä, kuten hiiren ja tavaraluettelon käyttöliittymät. Esimerkiksi hiirelle voidaan määritellä käskyt ”katso”, ”ota” ja ”kävele” ja kursorit niille. Kursorit tehdään *Cursors*-näkymässä ja ne voidaan

myös animoida. Tavaraluettelon käyttöliittymään voidaan määritellä kohdat, joihin eri tavarat sijoitetaan, kun ne on otettu. [68]

Hahmoille voidaan lisätä käyttöliittymiä, kuten hiiri ja tavaraluettelo. Hahmolle voidaan lisätä yleisiä kommentteja, joita hän sanoo satunnaisesti, kun suoritetaan käsky, joka ei varsinaisesti tee mitään. Esimerkiksi jos pelaaja yrittää liittää yhteen kaksi tavaraluettelon tavaraa, joita ei ole tarkoitus eikä voida pelissä liittää yhteen, pelaajahahmo voi kommentoida jotain yleistä, kuten ”En tiedä, miten nämä voidaan yhdistää”. Jotta pelaajahahmo ei sanoisi aina täsmälleen samaa, voidaan tehdä kommenttijoukkoja, joista kommentti valitaan satunnaisesti. [68]

Scripts-näkymässä voidaan kirjoittaa skriptejä. Skriptejä voidaan kutsua muun muassa edellä kuvatuista hahmojen, tavaroiden ynnä muiden toiminnoista. Skripteissä voidaan käyttää ohjelmassa määriteltyjä instansseja, kuten ruutuja, hahmoja ja tavaroita. Kielenä on Lua-skriptikielen ja Visionare Object Model -mallin yhdistelmä [69]. Visionare Object Model -mallilla päästään käsiksi ohjelmassa luotuihin instansseihin [69]. Lisäksi Visionairessa on partikkelijärjestelmä. Sillä voidaan luoda esimerkiksi tulen tai sateen näköisiä tehosteita. [68]

4.4 Pelimoottorien tallennusmuodot ja suunnitelman vienti

AGS tallentaa pelin objekteja ja asetuksia XML-muodossa .agf-päätteiseen tiedostoon. XML-tiedostossa on työkalun kannalta oleellisina tagit *Rooms, InventoryItems, Sprites* ja *Scripts*. Näiden alla on elementit objekteille, joita työkalun suunnitelmassa määritellään. Työkalun jatkokehityksessä oleellisia ovat myös tagit *Characters, Dialogs* ja *Cursors*. Ohjelmassa 4.2 on esimerkkinä *InventoryItem*. Sillä on tunniste, hotspot-kohdan koordinaatit, tieto, aloittaako pelaaja omistaen tavaran, kuvaus ja nimi. Kuvissa on vain viittaus *Sprite*-kuvan tunnisteeseen. Myöskään *Sprite*-elementissä ei ole polkua kuvaan. AGS-editorissa kuva tuodaan peliin *Sprite Manager* -komponentin kautta [70]. Spritet pitäisi myös tuoda ohjelmallisesti pelimoottoriin. Lisäksi *InventoryItem*-elementissä on linkit, toimintoihin, joita objektiin liittyy. Tapahtumalle ”Interact inventory item” on määritelty funktio *iKey_Interact*, joka suoritetaan, kun tapahtuma tehdään. *InventoryItem*-elementin funktio on oletuksena toteutettu *GlobalScript*-skriptissä.

AGS-pelimoottorissa huoneet on tallennettu crm-tiedostomuodossa. Pelin XML-tiedostossa on listattu huoneet, jotka ladataan crm-tiedostoista. XML-tiedostossa on vain huoneen numero ja kuvaus. Huonehan sisältää myös muun muassa taustakuvan, käveltävät alueet, hotspot-kohteet, huoneessa olevia tavaroita ja asetuksia. Huoneiden sisältö olisi lisättävä ohjelmallisesti. AGS-editoriin voidaan tehdä liitännäisiä [71]. Sellaisen avulla olisi todennäköisesti mahdollista luoda huoneen sisältö. Komponentin tekeminen ei olisi enää yksinkertaista, koska jouduttaisiin opettelemaan AGS-editorin liitännäisten teko ja tarvittavat rajapinnat.

```
<InventoryItems>
  <InventoryItemFolder Name="Main">
    <SubFolders />
    <InventoryItems>
      <InventoryItem>
        <ID>1</ID>
        <HotspotX>0</HotspotX>
        <HotspotY>0</HotspotY>
        <PlayerStartsWithItem>False</PlayerStartsWithItem>
        <CursorImage>2</CursorImage>
        <Image>2</Image>
        <Description>Key</Description>
        <Name>iKey</Name>
        <Properties />
        <Interactions>
          <Event Index="0" />
          <Event Index="1">iKey_Interact</Event>
          <Event Index="2" />
          <Event Index="3" />
          <Event Index="4" />
        </Interactions>
      </InventoryItem>
    </InventoryItems>
  </InventoryItemFolder>
</InventoryItems>
```

Ohjelma 4.2: Adventure Game Studion XML-muoto tavaraluettelon tavaroille.

Myös Visionaire tallentaa peliprojektin objektit ja asetukset XML-muodossa, .ved-päätteiseen tiedostoon. Oleellisina ylätason tageina ovat *Scenes, Objects, Sprites* ja *Scripts*. Jatkokehityksessä oleellisia voivat olla *Characters, Dialogs, Cursors, Interfaces, InterfaceClasses* ja *CommentSets*. Myös Visionairessa kuvat on tallennettu *Sprite*-elementtiin, mutta elementissä on tallennettuna tiedostopolku. Visionairessa kuvia ei tarvitse erikseen tuoda ohjelmaan ennen kuin niitä voidaan liittää erilaisille objekteille. Objektin kuva voidaan määrittää suoraan tiedostopolkuna.

Ohjelmassa 4.3 on esitetty esimerkki Visionairen *Objects*-elementistä. *Object*-elementit voivat olla mitä tahansa objekteja: esimerkiksi huoneessa oleva tavara, valikon nappi tai tavaraluettelon tavara. Tavaraluettelon tavaroilla attribuutti *ObjectIsItem* on tosi. Nämä näkyvät Visionairessa *Items*-näkymän alla, kun muut objektit ovat ruuduissa (sceneissä). Ohjelmassa 4.3 on ensin ruudulla oleva objekti ”Rock”, jonka *ObjectIsItem*-arvo on epätosi, ja toisena *Items*-näkymän objekti ”Cake”, jonka *ObjectIsItem*-arvo on tosi. Rock-objektilla on polygoni, joka kuvaa sen peittämää aluetta ruudussa. Rock-objektilla on myös viite toimintoon, joka on määritelty *Actions*-tagin alla. *Action*-elementissä on määritelty, milloin toiminto suoritetaan. *Action*-elementissä on linkki yhteen tai useampaan *ActionPart*-elementtiin, joka määrittelee, mikä toiminto on kyseessä. *ObjectPosition* määrittelee, missä kohdin ruutua pelaajahahmo suorittaa toimintoja objektille.

Koska Visionairessa ja AGS:ssä pelin tiedot on tallennettuna XML-muodossa, myös työkalu luo suunnitelmasta XML-muotoisen tiedoston. Tiedoston avulla voidaan luoda automaattisesti runko pelille pelimoottorissa. Työkalu luo geneerisen XML-tiedosto-

```
<Objects newId="6" versionedId="-1" tableId="6">
  <Object name="Rock" id="4" order="0" lastModified="-1"
          ObjectDirection="-1" ObjectCenter="590" ObjectIsItem="F"
          ObjectConditionNegate="F" ObjectIsWalkable="F"
          ObjectScrollFactorX="100" ObjectScrollFactorY="100">
    <ObjectName parentLink="T" id="11" tableId="14" LinkAny="F"/>
    <ObjectPolygon>
      <point x="1310" y="138"/>
      <point x="1368" y="626"/>
      <point x="1250" y="648"/>
      <point x="994" y="584"/>
    </ObjectPolygon>
    <ObjectCondition parentLink="F" id="-1" tableId="-1" LinkAny="F"/>
    <ObjectSprite parentLink="T" id="8" tableId="13" LinkAny="F"/>
    <ObjectAnimation parentLink="F" id="-1" tableId="-1" LinkAny="F"/>
    <ObjectActions>
      <Link parentLink="T" id="6" tableId="7" LinkAny="F"/>
    </ObjectActions>
    <ObjectConditions></ObjectConditions>
    <ObjectAnimations></ObjectAnimations>
    <ObjectValues></ObjectValues>
    <ObjectPosition x="1220" y="675"/>
    <ObjectParticleSystem parentLink="F" id="-1" tableId="-1"
                          LinkAny="F"/>
    <ObjectSnoopAnimation parentLink="F" id="-1" tableId="-1"
                          LinkAny="F"/>
    <ObjectSnoopAnimationPos x="0" y="0"/>
  </Object>
  <Object name="Cake" id="5" order="0" lastModified="-1"
          ObjectDirection="-1" ObjectCenter="-1" ObjectIsItem="T"
          ObjectConditionNegate="F" ObjectIsWalkable="F"
          ObjectScrollFactorX="100" ObjectScrollFactorY="100">
    <ObjectName parentLink="T" id="14" tableId="14" LinkAny="F"/>
    <ObjectPolygon></ObjectPolygon>
    <ObjectCondition parentLink="F" id="-1" tableId="-1" LinkAny="F"/>
    <ObjectSprite parentLink="T" id="9" tableId="13" LinkAny="F"/>
    <ObjectAnimation parentLink="F" id="21" tableId="9" LinkAny="F"/>
    <ObjectActions></ObjectActions>
    <ObjectConditions></ObjectConditions>
    <ObjectAnimations>
      <Link parentLink="T" id="21" tableId="9" LinkAny="F"/>
    </ObjectAnimations>
    <ObjectValues></ObjectValues>
    <ObjectPosition x="0" y="0"/>
    <ObjectParticleSystem parentLink="F" id="-1" tableId="-1"
                          LinkAny="F"/>
    <ObjectSnoopAnimation parentLink="F" id="-1" tableId="-1"
                          LinkAny="F"/>
    <ObjectSnoopAnimationPos x="0" y="0"/>
  </Object>
</Objects>
```

Ohjelma 4.3: Visionairen XML-muoto pelin objekteille.

muodon, joka ei sinällään ole käytettävissä missään pelimoottorissa. Väliin tarvitaan komponentti, joka syöttää tiedoston tiedot kyseisen pelimoottorin vaatimaan muotoon. Jatkossa käytetyimpiin pelimoottoreihin vievät komponentit voisivat olla osa työkalua, jolloin työkalusta saadaan luotua suoraan runko näihin pelimoottoreihin.

Tästä saadaan vastaus toiseen tutkimuskysymykseen: Käytetään suunnittelutyökalua suunnitelman laatimiseen ja generoidaan suunnitelmasta XML-tiedosto. Lisäksi luodaan pelimoottorikohtainen komponentti, joka syöttää suunnitelman XML-tiedoston tiedot pelimoottorin käyttämään XML-tiedostoon.

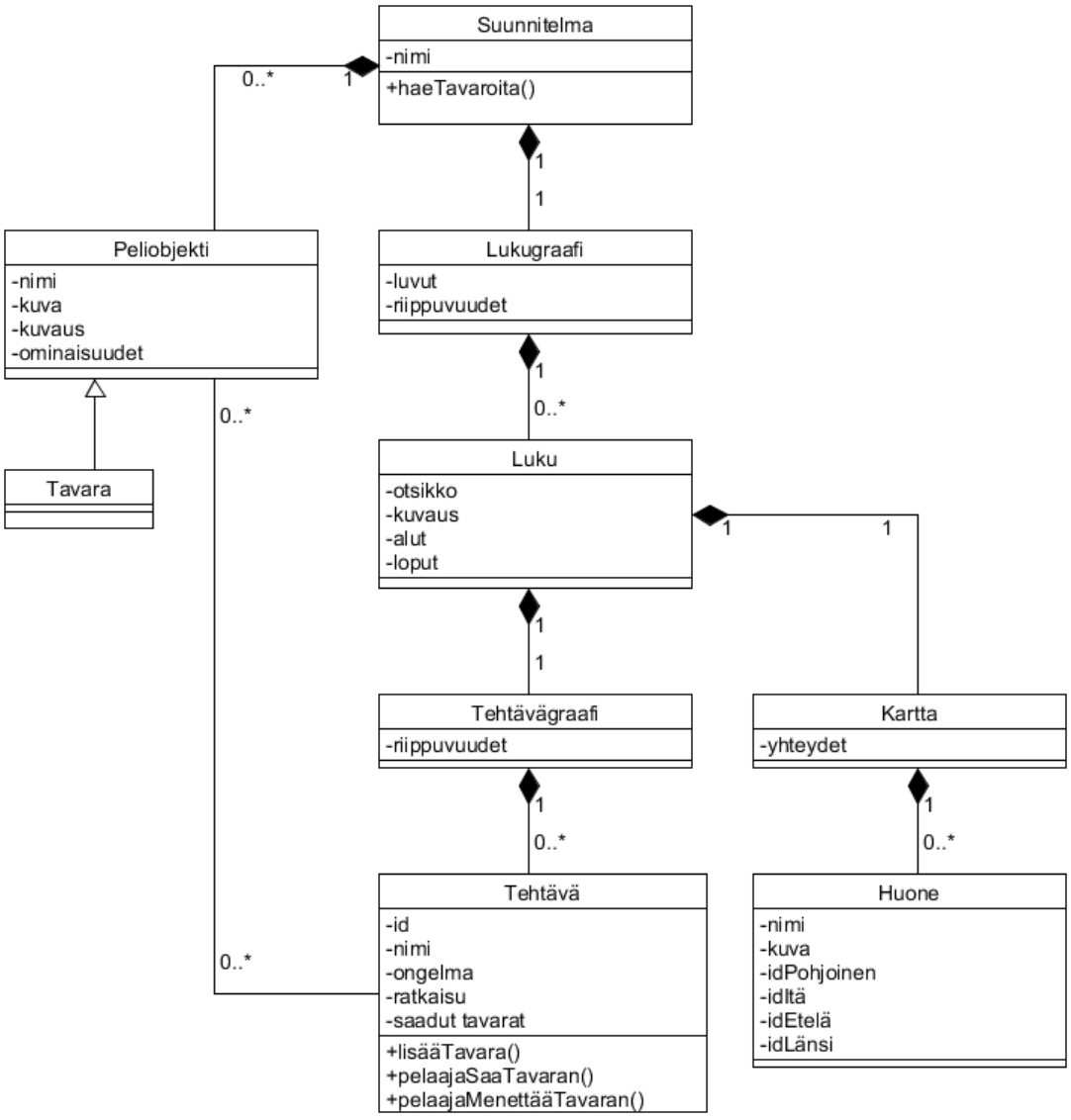
Työkalun generoimaan XML-tiedostoon tallennetaan kaikki suunnitelman tieto. Pelimoottoreihin kannattaa viedä tiedostosta kuitenkin vain huoneet ja tavarat. Tehtäviä ja tarinan kulkua ei ole määritelty AGS-pelimoottorissa minkäänlaisena valmiina rakenteena ja objekteina, vaan ne luodaan puhtaasti skriptaamalla. Visionairessa toimintoja voidaan luoda valmiilla elementeillä, mutta siinäkään ei ole mitään työkalua, jolla luodaan pelin kulku tai tehtävien riippuvuuksia. Skriptien generointi olisi monimutkaista ja ohjelmoija joutuisi todennäköisesti kuitenkin muokkaamaan niitä. Tieto on kuitenkin tallessa XML-tiedostossa, jos pelikehittäjä kuitenkin haluaa generoida skriptejä automaattisesti, tai jos joku seikkailupelimoottori myöhemmin tukee tarinan rakenteen syöttämistä XML- tai muussa muodossa.

5. TYÖKALU

Diplomityön osana toteutettiin prototyyppi seikkailupelien suunnittelutyökalusta. Se on esitelty tässä luvussa. Lisäksi sen toteutus ja käytetyt teknologiat on esitelty lyhyesti.

5.1 Toteutus

Luvussa 3 on määritelty ominaisuuksia, joita työkaluun tarvitaan. Kuvassa 5.1 on esitetty ominaisuuksien pohjalta luotu luokkakaavio työkalun tietomallista. Suunnitelma-



Kuva 5.1: Luokkakaavio työkalun tietomallista

luokka kuvastaa yhtä projektia eli yhden pelin suunnitelmaa. Luvut esitetään graafissa. Suunnitelmalla on Lukugraafi-olio, joka muodostuu Luku-olioista. Tehtävägraafi ja kartta ovat lukukohtaisia eli Luku-olio muodostuu niitä vastaavista olioista. Tehtävägraafi muodostuu Tehtävä-olioista ja määrittää niiden riippuvuudet. Kartta muodostuu Huone-olioista ja kuvaa niiden yhteydet toisiinsa. Tavarat tallennetaan Suunnitelma-olioon, koska niitä voidaan luoda irrallaan tehtävistä. Tehtävään voidaan liittää peliobjekteja. Peliobjekteista on tarkoitus nähdä, missä tehtävissä niitä käytetään, mutta sitä ei ole toteutettu vielä prototyyppiversioon.

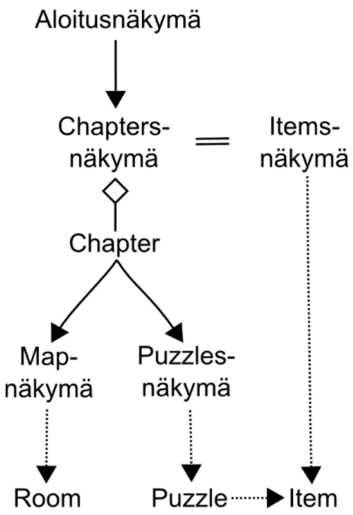
Tavara-luokka periytettiin jatkokehitystä ajatellen Peliobjekti-luokasta. Tavara-luokassa ei tosin toistaiseksi ole mitään Peliobjekti-luokasta eroavaa, joten tavarat voisivat olla suoraan pelkkiä peliobjektejakin. Peliobjekti-luokasta oli tarkoitus periyttää myös Hahmo-luokka, jolle voidaan lisätä dialogeja. Jatkokehityksessä peliobjektista ei periytetäkään mitään, vaan tavarat ja hahmot ovat suoraan peliobjekteja (ks. 6.2.4).

Käyttöliittymän toteutuksessa käytettiin Qt:n Graphics View Framework -kehystä [72]. Siinä QGraphicsView visualisoi QGraphicsScene-olion ja sen sisältämät QGraphicsItem-oliot. Luku- ja tehtävägraafien sekä kartan piirtoa varten toteutettiin mukautetut luokat QGraphicsScene-luokasta. Luvun, tehtävän ja huoneen sekä niiden yhteyksien esittämiseen toteutettiin mukautetut luokat QGraphicsItem-luokasta.

5.2 Työkalun esittely

Prototyypissä ei ole aikataulurajoitusten vuoksi toteutettu kaikkia ominaisuuksia, jotka tässä esitellään. Sen sijaan esitetään, millainen työkalun ensimmäinen versio tulee olemaan. Esimerkiksi ominaisuudet, jotka ovat hyvin samankaltaisia kuin jo toteutetut ominaisuudet, on voitu jättää pois, ellei niiden käyttö ole välttämätöntä työkalun testaamiselle. Prototyyppi on kokeilukelpoinen, mutta projektia ei voi esimerkiksi tallentaa ja ladata, jotta sitä voisi työstää useamman istunnon ajan, joten sitä ei olisi mielekästä käyttää oikeassa projektissa tässä muodossa. Työkalun kehitystä aiotaan jatkaa diplomityöhön tehdyn version jälkeen.

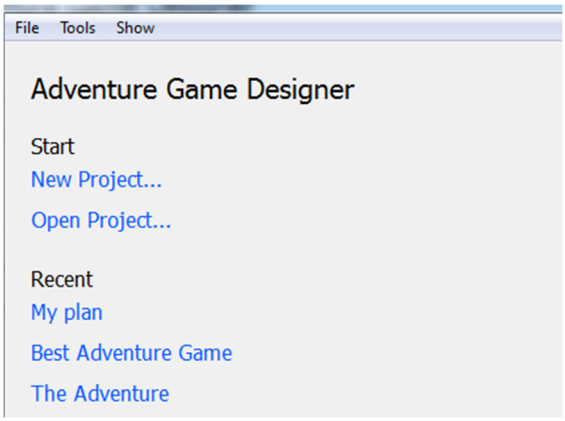
Työkalu koostuu viidestä päänäkymästä: aloitusnäkymä, Chapters-näkymä, Puzzles-näkymä, Map-näkymä sekä Items-näkymä. Tavaralla, huoneella ja tehtävällä on niiden ominaisuudet esittävät näkymät, jossa kyseistä objektia voidaan myös muokata. Niitä kutsutaan jatkossa lisänäkymiksi. Kuvassa 5.2 on esitetty navigointi työkalussa. Chapters- ja Items-näkymiin voidaan navigoida mistä vain navigointipalkin nappien kautta. Chapters-näkymässä näytetään luodut Chapter-elementit. Chapter-elementistä voidaan navigoida Map-näkymään kyseisen luvun karttaan sekä Puzzles-näkymään tehtävägraafiin. Map-näkymässä valitsemalla huone aukeaa Room-lisänäkymä. Vastaavasti Puzzles-näkymässä saadaan esiin Puzzle-lisänäkymä ja Items-näkymässä Item-lisänäkymä. Lisäksi Puzzle-lisänäkymän kautta saadaan esiin Item-lisänäkymä. Puzzles-näkymän navigointikohdat on esitetty liitteessä A.



Kuva 5.2: Navigointi työkalussa.

5.2.1 Projekti

Projekti tarkoittaa työkalussa yhtä pelisuunnitelmaa. Työkalu aukeaa aloitusnäkymään, jonka luonnos on esitelty kuvassa 5.3. Käyttäjä voi aloittaa uuden suunnitelman luomisen, tai avata aikaisemmin luotuja projekteja. Ylävalikoiden kautta projekti voidaan tallentaa ja ladata. Projektin nimeä voidaan muokata.



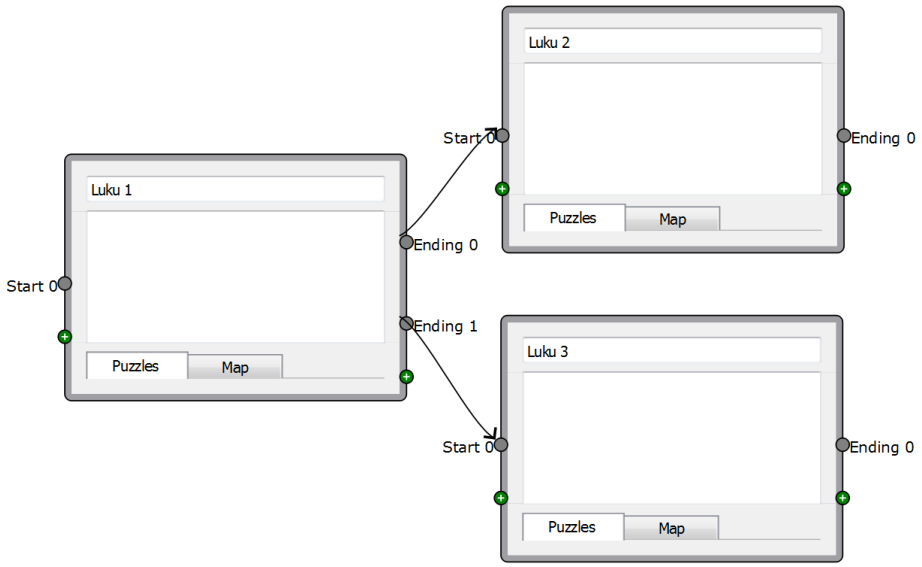
Kuva 5.3: Luonnos työkalun aloitusnäkymästä.

5.2.2 Chapters-näkymä ja luvut

Työkalun projekti aukeaa Chapters-näkymään. Käyttäjä lisää näkymään lukuja. Luvulla on vapaat tekstikentät nimi/otsikko ja kuvaus, johon voidaan kirjoittaa, mitä luvussa tapahtuu.

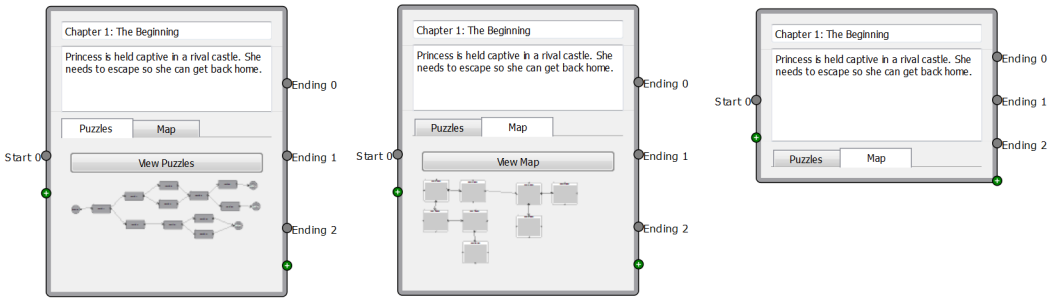
Luvulla on vähintään yksi alku ja loppu. Toisen luvun loppu voidaan yhdistää toisen luvun alkuun, jolloin saadaan kuvattua tarinan kulkua. Luvuille voidaan lisätä alkuja ja loppuja, jolloin voidaan suunnitella haarautuvia tarinanhaaroja. Esimerkiksi luvulla 1

voi olla loppu ”Ending 0”, joka on yhdistetty luvun 2 alkuun ”Start 0” ja loppu ”Ending 1”, joka on yhdistetty luvun 3 alkuun ”Start 0”. Tämä on esitetty kuvassa 5.4. Jos pelaaja päätyy luvun 1 loppuun ”Ending 1”, tarina jatkuu luvusta 3 ja siinä alkuasetelmasta ”Start 0”. Lukujen alut ja loput näkyvät myös luvun tehtävägraafissa, mikä esitetään tarkemmin alakohdassa 5.2.3.



Kuva 5.4: Lukujen yhteydet. Jokaisella luvulla on yksi tai useampi alku ja loppu. Luvun loppu voidaan yhdistää toisen luvun alkuun.

Chapters-näkymässä näytetään pienet kuvat tehtävägraafista ja kartasta, kuten kuvassa 5.5 näkyy. Niitä ei voida muokata Chapters-näkymässä. Tarkoituksena on, että käyttäjä näkee, ovatko eri lukujen tehtävägraafit ja kartat tasapainossa keskenään. Kuvat esitetään välilehtien alla. Jos näkymässä on paljon lukuja vierekkäin, kuvat voidaan piilottaa kokonaan, jotta luvut näkyisivät paremmin samaan aikaan.



Kuva 5.5: Luku työkalussa. Luvussa näkyy pieni kuva tehtävägraafista ja kartasta välilehtien alla. Ne voi myös piilottaa, jotta näkymään saadaan enemmän tilaa.

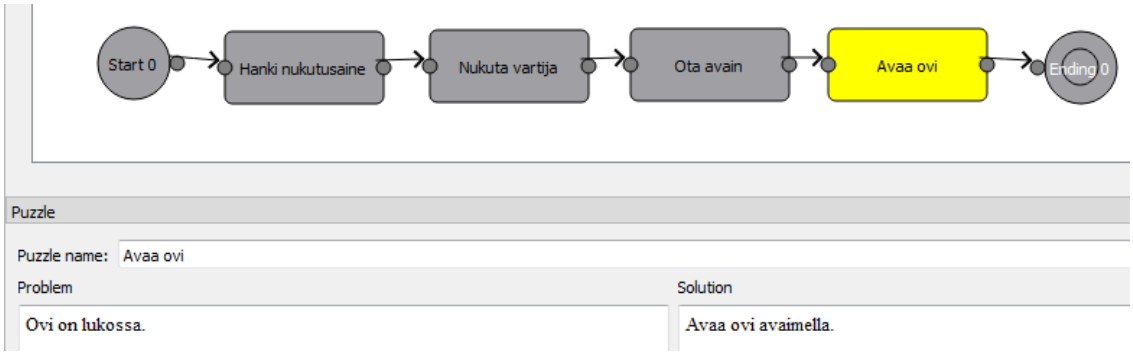
5.2.3 Puzzles-näkymä ja tehtävät

Jokaisella luvulla on oma tehtävägraafi. Se kuvaa tehtävät ja niiden väliset riippuvuudet. Tarkoituksena on esittää tehtävät ja niiden riippuvuudet visuaalisessa muodossa, jotta ne on helpompi hahmottaa ja huomata pullonkaulat sekä umpikujat.

Aluksi tehtävägraafissa näkyy aloitukset ja lopetukset, jotka kyseiselle luvulle on Chapters-näkymässä lisätty. Tehtävägraafilla on siis vähintään yksi aloitus ja yksi lopetus. Alku voidaan yhdistää tehtävään tai tehtäviin, jotka sitä seuraavat. Eri alkuasetelmista pelaaja voi siis päästä erilaisiin tehtäviin luvussa. Alkuasetelma voi myös määrätä, mitä tavaroita pelaajalle on jäänyt edellisestä luvusta. Pelaajalla on siis käytössään eri tavarat riippuen siitä, mihin lukuun ja lopetukseen aloitus on liitetty.

Tehtävägraafissa tehtäville on tarkoitus kirjoittaa vain lyhyt kuvaus, jotta tehtävien kulkua voidaan tarkastella helposti yleistasolla ja ne mahtuvat mahdollisimman hyvin näkyviin kerrallaan. Jokaisella tehtävällä on sisään- ja ulostuloja, joiden avulla tehtäville luodaan riippuvuuksia. Jokaisella tehtävällä on yksi sisään- ja ulostulo, joihin voidaan yhdistää useampia riippuvuusnuolia.

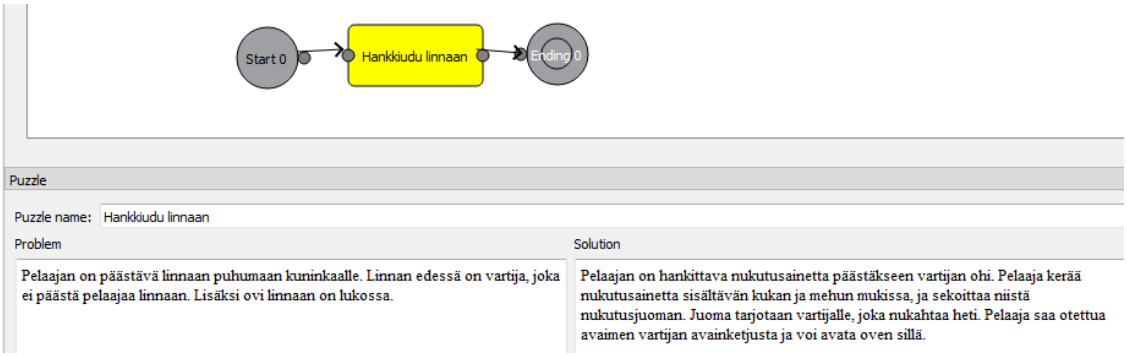
Valitsemalla tehtävä voidaan tarkastella sen lisänäkymää ja muokata tehtävää. Tehtävä muodostuu ongelmasta ja ratkaisusta. Käyttäjä voi kirjoittaa ongelman ja ratkaisun vapaasti tekstikenttiin. Käyttäjä voi päättää, millä tasolla kirjoittaa tehtävät. Tehtävägraafissa näkyvät tehtäväkuvaukset voivat olla hyvin tarkalla tasolla, kuten kuvassa 5.6, esimerkiksi tehtävä ”Avaa ovi”, jota varten täytyy näpistää avain nukkuvan vartijan taskusta. Tällöin tehtävän ongelma- ja ratkaisukenttiin tuskin tulee paljoa tekstiä, ellei käyttäjä halua kuvata muitakin asioita kuin toimia tekstikentässä, kuten motiiveja tai vaikka tehtävään liittyvää ympäristöä ja hahmoja.



Kuva 5.6: Kuva työkalusta. Linnaan pääseminen koostuu useasta tehtävästä. Tehtävät koostuvat vain yhdestä toiminnosta.

Toisaalta tehtävägraafin kuvaus voi olla yleisemmällä tasolla. Yllä esitetyn tehtävän lyhyt kuvaus olisi ”Hankkiudu linnaan”. Se on esitetty kuvassa 5.7. Ongelma-kentässä kerrotaan, että pelaajan tarvitsee päästä linnaan, mutta oven edessä on vartija ja ovi on myös lukossa. Ratkaisukentässä kerrotaan, että pelaajan on hankittava nukutusainetta,

käytettävä sitä vartijaan ja sen jälkeen otettava nukkuvalta vartijalta avain ja avattava ovi sillä.

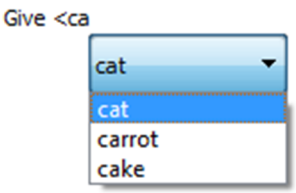


Kuva 5.7: Kuva työkalusta. Tehtävä sisältää monta toimintoa, jotka on kuvattu ratkaisussa.

Käyttäjä voi käyttää tiettyä syntaksia lisätäkseen tavaroita ongelma- ja ratkaisuteksteihin. Jos käyttäjä kirjoittaa ”There is a <key> hidden”, työkalu luo automaattisesti tavaran ”key” ja lisää sen tehtävän tavaroiden listaukseen kuin myös kaikkien tavaroiden listaukseen. Kirjoitettu objekti muuttuu napiksi ja syntaksi ei jää näkyviin, koska se voisi häiritä tekstin luettavuutta. Napista pääsee kyseisen tavaran lisänäkymään, jossa sille voidaan lisätä tietoja sekä muokata tavaraa. Tavarat on myös helppo huomata tekstistä, kun ne ovat erinäköiset kuin muu teksti. Kuvassa 5.8 on esimerkki ratkaisusta, johon on liitetty tavarat ”carrot” ja ”rabbit”.

Kuva 5.8: Tehtävän ratkaisu, joka sisältää tavarat carrot ja rabbit.

Kun tavaraa aletaan kirjoittaa syntaksin mukaisesti, työkalu tarjoaa jo olemassa olevia tavaroita, sekä niiden nimen että lisättyjen ominaisuuksien perusteella. Tämä auttaa uudelleenkäyttämään muissakin tehtävissä tarvittuja tavaroita. Kuvassa 5.9 on esitetty tavaroiden ehdotus.



Kuva 5.9: Tavaroiden ehdotus tehtävän ongelma- ja ratkaisukentissä

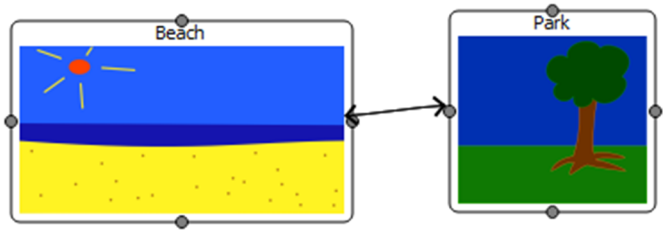
Jos syntaksia halutaan käyttää, työkalua tulisi käyttää englanniksi tai muulla kielellä, jossa sanat eivät taivu. Esimerkiksi suomen kielellä suora lisääminen syntaksin avulla ei toimi niin hyvin, koska sanoja taivutetaan. Esimerkiksi ”<Maton> alla on <avain>” loisi

objektin ”maton” objektin ”matto” sijaan. Jos myöhemmin halutaan tukea muita kieliä, syntaksiin voidaan lisätä tuki sanojen taivutukselle.

Tarkassa tehtävänäkymässä on listattuna tavarat, jotka tehtävään on lisätty. Käyttäjä voi merkata tavaran kohdalle, että pelaaja saa sen kyseisessä tehtävässä tavaraluetteloonsa. Tällöin tehtävää seuraaviin tehtäviin tulee näkyviin, että tavara on käytettävissä tehtävässä. Tämä auttaa varmistamaan, että tavarat ovat saatavilla, kun niitä tarvitaan. Lisäksi suunnittelija voi saada uusia ideoita, miten tehtävä ratkaistaisiin, kun hän näkee, mitä tavaroita on jo saatavilla. Jos tavara käytetään tai muuten menetetään tehtävässä, se voidaan merkitä menetetyksi, jolloin se ei ole enää käytettävissä olevien listassa kyseistä tehtävää seuraavissa tehtävissä. Samalla tavalla luvun lopetuksessa valitaan, mitkä tavarat siirtyvät lopetukseen liitettyyn toisen luvun aloitukseen.

5.2.4 Map-näkymä ja huoneet

Karttanäkymässä voidaan lisätä paikkoja, eli huoneita, karttaan. Jokaisella luvulla on oma karttansa. Huoneita voidaan liittää toisiinsa ylä-, ala- ja sivureunoista, jotka kuvastavat ilmansuuntia, joihin ruudusta pääsee. Yhteen reunaan voidaan kuitenkin liittää useampi yhteys, sillä peleissä on usein myös muita vaihtoehtoja edetä seuraaviin ruutuihin kuin vain ruudun reunat. Yhteys voi olla yksisuuntainen tai kaksisuuntainen, sen mukaan pääseekö huoneesta takaisin toiseen huoneeseen vai ei. Kuvassa 5.10 on kaksi huonetta, joiden välillä on kaksisuuntainen yhteys.



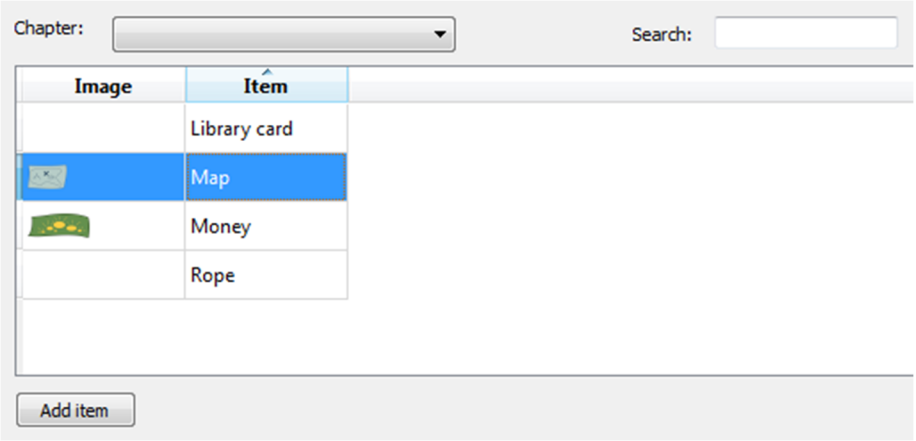
Kuva 5.10: Kaksi huonetta, joiden välillä pelaaja voi liikkua edestakaisin.

Huoneen lisänäkymässä voidaan asettaa huoneen nimi sekä kuva. Kuva näkyy myös karttanäkymässä.

5.2.5 Items-näkymä ja tavarat

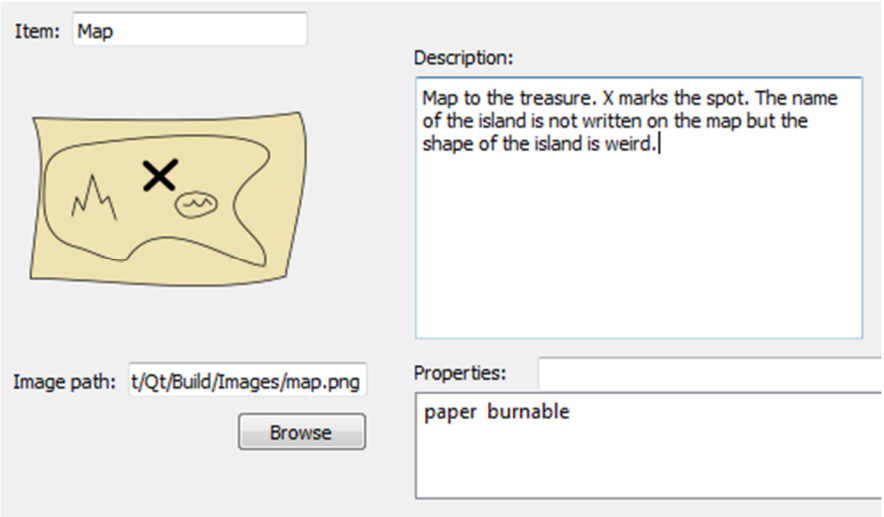
Items-näkymässä on listattuna kaikki peliin luodut tavarat. Kuvassa 5.11 on esitetty Items-näkymä. Käyttäjä voi luoda uusia tavaroita näkymässä. Valitsemalla tavara listasta päästään muokkaamaan sen tietoja Item-lisänäkymään.

Item-lisänäkymä on esitetty kuvassa 5.12. Liitteessä B on esitetty Items-näkymä ja Item-lisänäkymä kokonaisuudessaan työkalussa. Tavaralle määritellään nimi, tiedosto-



Kuva 5.11: Items-näkymän lista kaikista luoduista tavaroista.

polku kuvaan sekä kuvaus tavarasta vapaaseen tekstikenttään. Kuva voi olla suuntaa antava konseptitaidekuva tai varsinainen peliin tuleva kuva. Yksi käyttömahdollisuus on, että suunnittelija lisää kuvaksi luonnoksen siitä, millaisen on ajatellut tavaran olevan. Graafikko voi sitten luoda valmiin kuvan ja vaihtaa sen luonnoksen tilalle. Näin hän näkee mitä kuvia on tehtävä, ja myös mitkä hän on jo tehnyt.



Kuva 5.12: Item-lisänäkymässä voidaan muokata tavaran nimeä, kuvaa, kuvausta ja ominaisuuksia.

Kuvaukseen voidaan kirjoittaa esimerkiksi, millainen tavara on ja mitä sillä voidaan tehdä. Kuvaukseen voidaan myös kirjoittaa, mitä pelaajahahmon tulisi sanoa, kun hän katsoo tavaraa tai kohdetta. Lisäksi tavaralle voidaan liittää ominaisuuksia. Käyttäjä voi esimerkiksi lisätä tavaralle ”bucket”, ämpäri, ominaisuudet ”plastic”, eli ämpäri on muovia, ja ”can carry liquid”, sillä voidaan kantaa nesteitä. Ominaisuuksien avulla suunnittelija voi löytää tavaroita uudelleenkäytettäväksi toisissa tehtävissä. Luotuja tavaroita voidaan hakea nimellä tai ominaisuuksien perusteella. Käyttäjä voi myös valita tietyn luvun, jonka tavaroita haluaa tarkastella. Lisäksi työkalu näyttää, missä tehtävissä tavara on käytössä.

Käyttäjä voi luoda hotspot-kohteen kuten minkä tahansa tavaran, mutta ei merkitse tavaraa ikinä tehtävässä saatavaksi. Toki voisi olla selkeyden kannalta hyödyllistä, että nämä on eroteltu. Toisaalta ominaisuuksiin voidaan lisätä sana ”hotspot”, jos niin halutaan.

5.2.6 XML-tiedoston generointi

Tools-valikon alla on valinta ”Generate XML”. Siitä aukeaa tiedostovalikko, jossa käyttäjä määrittää generoitavan tiedoston nimen sekä tallennuspaikan. Muita asetuksia ei tehdä.

Työkalu luo geneerisen tiedoston, josta on esimerkki liitteessä C. Työkalu generoi kaikki elementit, joita pelissä on: luvut, tehtävät, kartan huoneineen ja tavarat. Näihin liittyy tekstimuotoista dataa, esimerkiksi luvulla on nimi ja kuvaus, tai huoneella nimi ja tiedostopolku kuvaan. Tavaralla on lista ominaisuuksia. Luvun aluilla ja lopuilla on tunnisteet, joiden avulla määritellään niiden yhteydet, eli mistä lopetuksesta mennään mihinkin alkuun. Samalla lailla tehtävillä ja kartoilla on tunnisteet, joiden avulla määritellään niiden yhteydet. Tehtäviin on merkitty tavarat, jotka on mainittu tehtävässä. Tehtävän tavaralla on boolean-arvo, joka määrittää, saako pelaaja tavaran tehtävässä vai onko tavara hotspot-kohde.

5.3 Valitut teknologiat toteutuksessa

Aikataulu, budjetti, alustariippumattomuus ja Unity-pelimoottorin käytön riskit huomioon ottaen työkalu päätettiin toteuttaa Qt-kehyksellä [73]. Qt:ssa kielenä on C++. Työkalu on työpöytäsovellus. Työkalun toteuttamiseen pohdittiin lähinnä vaihtoehtoja Qt-kehitysympäristö, .NET-kehys Windows Presentation Foundation -järjestelmän (WPF) kanssa tai Unity-pelimoottorin liitännäisenä (plugin) tai laajennoksena (extension).

Qt:ssa on hyvää, että se on alustariippumaton [74] ja työkalun tekijälle jossain määrin tuttu, jolloin asioiden opettelu ei vie liikaa aikaa toteutukselta. WPF on vain Windows-käyttöjärjestelmälle [75]. Sen käyttöä kokeiltiin, mutta todettiin, että sen opettelu veisi liian paljon aikaa ja laatu olisi todennäköisesti silti huonoa, koska kyseessä olisi työkalun tekijän ensimmäinen sillä tehty ohjelma.

Unity-pelimoottoria käytetään paljon monenlaisten pelien tekemiseen [76]. Unity-pelimoottorin liitännäiset ovat käytössä vain Unity-pelimoottorin pro-versiossa [77], joka maksaa 1500 dollaria tai 75 dollaria kuussa [78], joten sen käyttöä ei diplomityön puitteissa pidetty vaihtoehtona. Työkalun tekijällä on kuitenkin vain hyvin vähän kokemusta Unity-pelimoottorin käytöstä, eikä lainkaan kokemusta sen laajentamisesta. Toisaalta sen opettelu olisi työkalun tekijälle erittäin hyödyllistä, koska sitä käytetään pelialalla paljon. Lisäksi lisäosien jakaminen Unity-pelimoottorin Asset Store –palve-

lussa ja lisääminen Unity-pelimoottoriin on käyttäjän näkökulmasta yksinkertaista, joten jo Unity-pelimoottoria käyttävät voisivat ottaa työkalun helposti käyttöönsä. Seikkailupelejä tehdään kuitenkin paljon muilla pelimoottoreilla, vaikka Unity-pelimoottoriinkin on tarjolla seikkailupelien tekemiseen tarkoitettuja työkaluja (ks. 4.1). Jos suunnitelma tehdään Unity-pelimoottorilla, olisi kuitenkin luontevaa myös tehdä itse peli sillä, joten muita pelimoottoreita käyttävät saattaisivat rajautua pois käyttäjäkunnasta. Unity-pelimoottorin laajennoksen käyttäminen saattaa myös rajoittaa työkalun toteutusta. Unity-editori on tietynlainen ja työkalun tulisi taipua sen muotoon, jolloin työkalusta voisi tulla liian monimutkainen lista asetuksia.

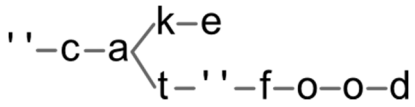
Työkalua lähdettiin suunnittelemaan piirtämällä käyttöliittymäkuvat paperille sekä kirjoittamalla määrittelydokumenttia. Lisäksi piirrettiin luokkakaavio selventämään, mistä objekteista työkalu koostuu ja miten ne liittyvät toisiinsa. Määrittelydokumenttia kirjoitettiin lähinnä, jotta työkalun ominaisuuksista saataisiin jonkinlainen kuva ennen toteutusta. Sen jälkeen siitä luovuttiin ja toteutus tehtiin hyödyntäen käyttöliittymäkuvia ja luokkakaaviota. Koska työkalua teki vain yksi ihminen, ei ollut välttämätöntä, että koko työkalu olisi dokumentoitu etukäteen ja jatkuvasti, kun muutoksia tulee. Myöhempää jatkokehitystä varten työkalun nykyinen tila tulee dokumentoida.

Työkalun koodi tallennettiin Bitbucket-palveluun [79] käyttäen SourceTree-ohjelmaa [80] ja sen pohjana Mercurial-versionhallintatyökalua [81]. Koska työkalua tehtiin yksin, päätarkoitus oli tallentaa koodi muuallekin kuin yhdelle koneelle. Tietokone kaatui kerran ohjelman tallentamisen ja kääntämisen aikana, jonka seurauksena yksi tiedosto pyyhkiytyi tyhjäksi. Viimeisin toteutus menetettiin, mutta aikaisempi versio otettiin pohjaksi, jotta koko luokkaa ei tarvinnut kirjoittaa uudelleen.

Työkalun toteutuksessa käytettiin lopulta vain yhtä valmista komponenttia. Työkalun toteutus aloitettiin käyttäen valmiita ulkopuolisen toteuttamia luokkia graafien piirtoon. Se kuitenkin huononsi ohjelman rakennetta ja olisi monimutkaistanut jatkototeutusta liikaa. Muitakaan sopivia komponentteja ei löytynyt. Työkalun toteutus aloitettiin uudelleen alusta ja graafien piirto toteutettiin itse. Työssä käytettiin työkalun tekijän aikaisemmassa projektissa toteuttamaa trie tree -rakennetta, joka sopii sanahaun toteuttamiseen. Siitä tarvitsi muuttaa vain trie tree -olioon tallennettavien osoittimien tyyppi. Lisäksi siihen lisättiin sanojen poisto trie tree -oliosta, mitä ei ollut toteutettu laisinkaan aikaisemmassa projektissa.

Työkalussa tavaroiden haku Problem- ja Solution-tekstikentissä on toteutettu trie tree -tietorakenteen avulla. Trie-tietorakenteessa on tallennettuna kaikki tavaroiden nimet ja ominaisuudet. Trie-rakenne muodostuu solmuista. Solmu edustaa yhtä kirjainta. Sana lisätään puuhun lisäämällä ensin ensimmäinen kirjain puun juuren lapsisolmuksi, ja sen jälkeen seuraavat kirjaimet aina edellisen lapsisolmuksi. Sanan viimeisen kirjaimen solmuun merkitään, että sana on olemassa, ja lisätään osoittaja tavaraan, jonka nimi tai ominaisuus sana on. Toinen sana, joka alkaa samalla tavalla kuin jo lisätty sana, käyttää

samoja ensimmäisiä solmuja ja haarautuu eri lapsisolmuihin, kun sana ei jatku samalla tavalla. Yhdessä solmussa voi olla osoittimet moneen tavaraan, koska tavaroilla voi olla samoja ominaisuuksia. Kuvassa 5.13 on esimerkki. Puuhun on lisätty tavaroiden nimet ”cake” ja ”cat” sekä ominaisuus ”cat food”, joka voisi osoittaa esimerkiksi tavaroihin kala ja hiiri.



Kuva 5.13: Esimerkki trie tree -rakenteesta. Puussa on sanat cake, cat ja cat food.

6. ARVIOINTI JA JATKOKEHITYS

Luvussa arvioidaan työkalun ominaisuuksia suunnittelemalla sillä testipeli. Lisäksi pohditaan, mitä muita uusia ominaisuuksia työkaluun kannattaa lisätä.

6.1 Testipeli ja työkalun arviointi

Työkalua testattiin testipelillä. Testauksen tarkoituksena on selvittää, auttaako työkalu seikkailupelin suunnittelussa. Tässä luvussa ei siis puhuta teknisestä toimivuudesta, vaan arvioidaan työkalun ominaisuuksia.

Testiseikkailupeliä ei voida suunnitella valmiiksi etukäteen, esimerkiksi paperilla, ja vain siirtää työkaluun, koska työkalun tarkoitus on auttaa suunnittelussa. Seikkailupeliä siis suunniteltiin samalla kun työkalua testattiin. Kuitenkin työkalun testausta varten määriteltiin testipelin ominaisuuksia, joita seikkailupelissä tulisi olla, jotta se testaa työkalun eri ominaisuuksia.

Testipelissä on yksi pelaajahahmo. Testiseikkailupeli jakaantuu lukuihin. Pelissä on haarautuva juoni, jotta voidaan testata lukujen haarautumista. Pelissä on useampia peräkkäisiä ja rinnakkaisia tehtäviä. Työkalu ei prototyyppiversiossaan tue vaihtoehtoisia ratkaisuja tehtäviin, joten niitä ei tehdä. Tehtävissä saadaan tavaroita, joita käytetään myöhemmissä tehtävissä. Tavaroille annetaan ominaisuuksia ja tarkastellaan, auttaako se niiden uudelleenkäyttämisessä. Luvuille tehdään kartat. Testipeli tehdään englanniksi, koska se on työkalun tukema kieli.

Seikkailupelin nimi on *Treasure Hunt*. Seuraavassa on esitelty pelin suunnittelua työkalun avulla samalla arvioiden työkalun ominaisuuksia.

6.1.1 Luvut

Testipeli koostuu neljästä luvusta, jotka on esitetty alla. Luvut 2.1 ja 2.2 ovat keskenään vaihtoehtoisia. Pelaaja päätyy vain toiseen niistä yhden pelin aikana. Vaihtoehtoisista luvuista päädytään kuitenkin samaan loppulukuun.

> ***Chapter 1: Treasure map***
>
> Guy (the player) inherits a treasure map from his departed mother. He will need to find out where the island on the map is, how to get there and get money to fly to an island nearby the treasure island. Once he arrives on the nearby island he needs to get some kind of a boat to get

to the treasure island because there are sharks in the sea. He can choose a raft or a giant leaf.

Chapter 2.1: Going to treasure island by raft

Guy arrives at the island on the west side where it is windy. He takes the treasure map out to read it but the wind blows it away from his hands. He will have to find it and get it back. He can go to a cove where it is not windy to read it.

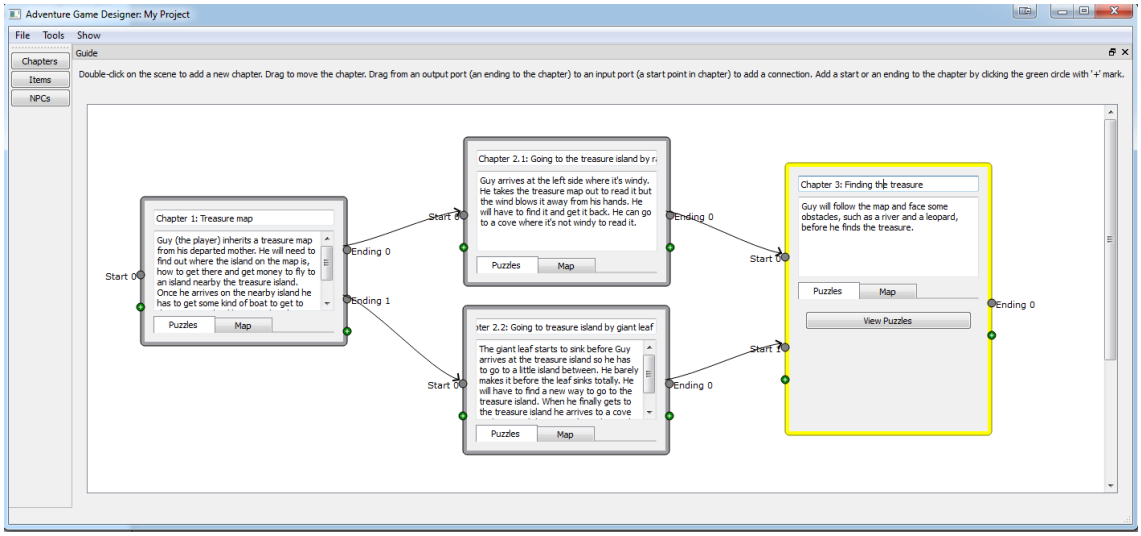
Chapter 2.2: Going to treasure island by giant leaf

The giant leaf starts to sink before Guy arrives at the treasure island so he has to go to a little island between. He barely makes it before the leaf sinks totally. He will have to find a new way to go to the treasure island. When he finally gets to the treasure island he arrives to a cove and can read the map without the wind blowing it away.

Chapter 3: Finding the treasure

Guy will follow the map and face some obstacles, such as a river and a leopard, before he finds the treasure.

Kuvassa 6.1 on Chapters-näkymä testipelin suunnitelmasta. Chapter-elementin tekstikenttä tarinaa varten on melko pieni. Siihen sopii ainoastaan lyhyt kuvaus siitä, mitä luvussa tapahtuu. Suunnittelija voi haluta kirjoittaa hyvinkin yksityiskohtaisen kertomuksen luvusta. Tällöin olisi hyvä, että kuvauksen saisi avattua uuteen isoon tekstinmuokkausikkunaan. Toisaalta Puzzles/Map-välilehtien piilotuksella saadaan lisää tilaa kertomukselle, mutta siitä huolimatta erillinen tekstinmuokkausmahdollisuus olisi tarpeellinen pitkien kuvausten tapauksessa. Chapters-näkymässä luvusta voisi kuitenkin

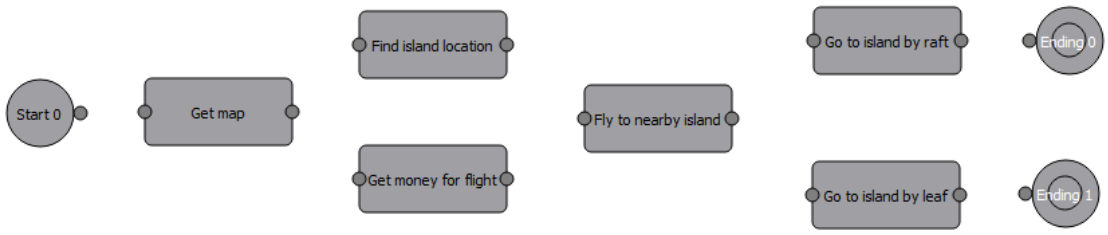


Kuva 6.1: Chapters-näkymä työkalussa testipelin suunnitelmasta. Testipeli koostuu neljästä luvusta. Toinen luku haarautuu kahteen mahdolliseen lukuun. Pelaaja päätyy yhden pelin aikana vain toiseen niistä valinnoistaan riippuen.

näkyä lyhyt kuvaus, josta saadaan yleiskuva, mitä luvussa tapahtuu. Kuvauksen vieressä olisi nappi, josta päästäisiin tekstinmuokkausnäkymään. Siellä voidaan kirjoittaa tarkka kuvaus luvusta. Luvun otsikosta saa myös helposti nopean muistikuvan, mistä luku kertoo.

6.1.2 Tehtävägraafi

Lukujen suunnittelun jälkeen siirrytään luvun tehtävägraafin muokkaukseen Puzzles-näkymään. Tällöin huomataan, että luvun kuvaus tulisi olla näkyvissä Puzzles-näkymässä. Tehtävien suunnittelu on hyvä aloittaa luvun pääkohdista, mikä on hankalaa, kun tarinaa ei nähdä samaan aikaan. Kuvassa 6.2 tehtävät on luotu luvun tarinan pääkohdista. Niistä lähdetään rakentamaan lisää tehtäviä. Työkalussa on käytettävä riippuvuusgraafia, mikä edistää sitä, että jokainen tehtävä on mielekkäästi osa tarinaa eikä irrallinen tehtävä.



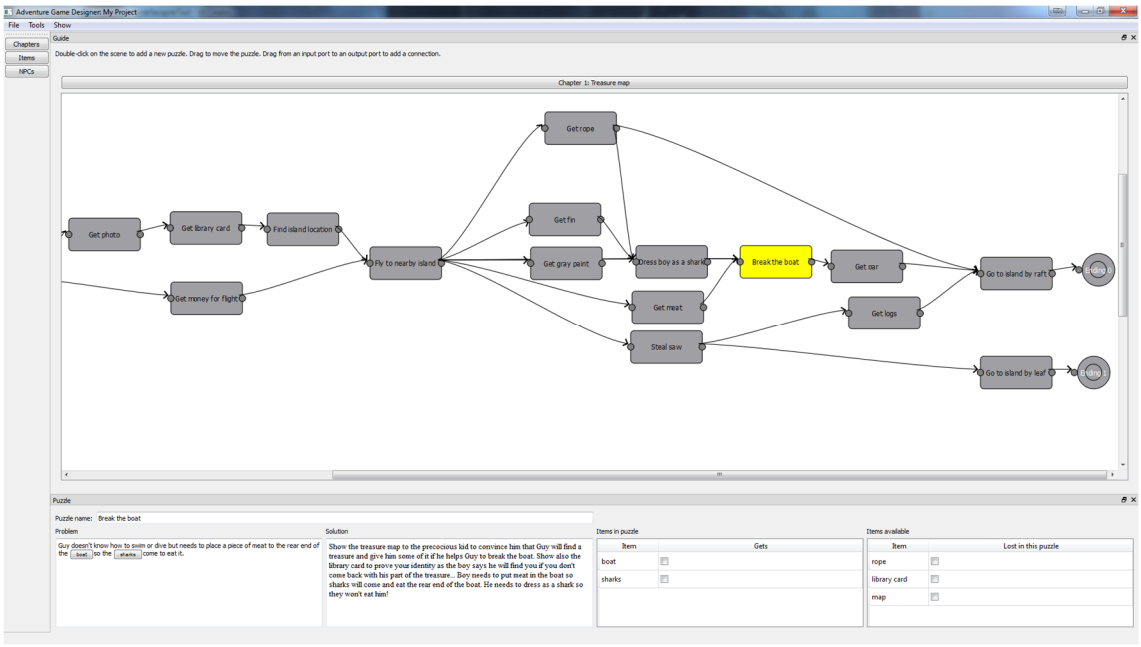
Kuva 6.2: Tehtävägraafin luonnos, johon on lisätty luvun tarinan pääkohdat.

Työkalun testatussa versiossa välianimaatiot voidaan luoda tehtävänä tehtävägraafiin. Selkeyden vuoksi olisi kuitenkin hyvä, että ne voitaisiin merkitä jotenkin erilliseksi tehtävistä. Tehtävän tarkassa näkymässä voisi olla valintaruutu, joka määrittää, että tehtävä onkin välianimaatio, ja muuttaa esimerkiksi tehtävän taustavärin tehtävägraafissa. Problem- ja Solution-kenttien sijaan voisi olla tekstikenttä, johon voidaan kirjoittaa kuvaus, mitä välianimaatiossa tapahtuu. Välianimaatio myös määrittelisi, mikä on lähtötilanne seuraaviin tehtäviin. Ilman sellaista alussa ensimmäinen tehtävä alkaa kuin tyhjästä.

Huoneetkin olisi hyvä voida luoda Problem- tai Solution-kentässä, esimerkiksi syntaksilla @huone@. Tällöin luvun karttaan luotaisiin huone annetulla nimellä. Karttanäkymässä voitaisiin sitten määrittää sen muita tietoja. Kun huoneet voidaan luoda vain karttanäkymässä, suunnittelijan pitää muistaa tai käydä tarkistelemassa tehtävistä, mitä kaikkia huoneita tarvitaan tehtävien toteuttamiseksi. Tämä on hyvin epäkäytännöllistä.

Kun tehtäviä tekee päätavoitteista taaksepäin, syntyy samalla luonnostaan lista tavaroita, joita kyseisen tehtävän ratkaisemiseen tarvitaan. Työkaluhan tarjoaa nyt vain tavaroita, joita on saatavilla seuraavissa tehtävissä. Se voisi myös näyttää edeltäville

tehtäville, että tällaisia tavaroita tarvitaan seuraavissa tehtävissä. Toisaalta tämä voidaan aivan hyvin toteuttaa niin, että tehdään heti tehtävät jokaisen tavaran hankkimiseksi, kun tavarat tulevat ilmi. Kuvassa 6.3 näkyy esimerkiksi testipelin tehtävä ”Go to island by raft”, jossa rakennetaan lautta. Lautan rakentamiseen tarvitaan köyttä ja tukkeja, sekä airo, että päästään eteenpäin ja oikeaan suuntaan. Kun tehtävää kirjoitettiin, mietittiin, mistä lautta rakennetaan ja kirjattiin heti ylös tehtävät ”Get rope”, ”Get oar” ja ”Get logs”. Vasta sen jälkeen lähdettiin miettimään, mistä kyseiset tavarat saataisiin. Tehtävien nimet voidaan muuttaa jälkikäteen nykyistä kuvaavammiksi, kun tiedetään, miten airo saadaan. Esimerkiksi tehtävälle ”Get oar” kuvaavampi nimi olisi ”Buy oar from fisher”. Liitteessä D on kuva koko ensimmäisen luvun tehtävägraafista.



Kuva 6.3: Osa testipelin ensimmäisen luvun tehtävägraafia työkalussa. Valittu tehtävä näkyy keltaisella, ja sen kuvaus lukee alareunassa.

Työkalulla on mahdollista tehdä tehtäviä päätavoitteista taaksepäin, mutta se ei mitenkään ohjaa siihen. Tehtävägraafin ansiosta työkalusta näkee hyvin, kuinka paljon tehtävissä on rinnakkaisia ja peräkkäisiä tehtäviä, ja että tehtävät ovat tasapainossa. Se ei kuitenkaan ohjaa tekemään monitasoisia ja tasapainoisia tehtävägraafeja. Guide-näkymässä neuvotaan, miten työkalua käytetään. Siinä voitaisiin antaa ohjeita myös hyvistä suunnittelumenetelmistä, joita työkalulla voidaan käyttää. Kaikkiin ongelmakohtiin seikkailupeleissä työkalu ei voi vaikuttaa, vaan ne ovat puhtaasti suunnittelijan vastuulla. Näistäkin voidaan antaa vinkkejä Guide-näkymässä, jos se koetaan tarpeelliseksi.

Jo melko yksinkertaisessa testipelissä tehtäviä jouduttiin jonkin verran järjestelemään sopivaan asetelmaan, jotta tehtävät ja tehtävien yhteydet eivät menisi päällekkäin tai epäloogiseen järjestykseen, mikä hankaloittaisi tehtävägraafiin lukemista. Työkaluun tarvitaan ominaisuus, joka asettelee tehtävägraafin automaattisesti. Myös Gilbert toteaa,

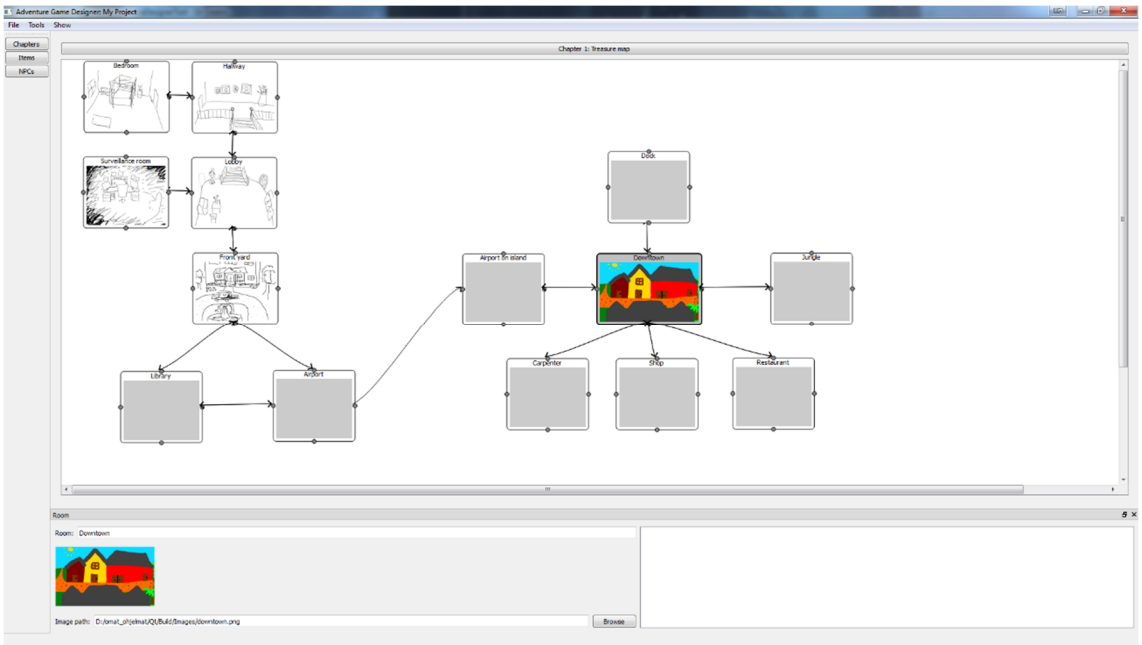
että tällainen ominaisuus on välttämätön Puzzle Dependency Chart -kaaviota tehtäessä [41].

Alakohdassa 5.2.3 esitetään, kuinka tehtävät voidaan kuvata eri tasoilla työkalussa: joko useampana tehtävänä tarkasti tehtävägraafissa tai yhden tehtävän sisällä ongelma- ja ratkaisukentissä. Jos työkalussa voitaisiin kuvata eri tasoja, työkalusta näkisi niin yleiskuvan tehtävägraafista kuin tarkemmin tehtävät ja niiden yhteydet. Toisin sanoen ylätason tehtävän alle voitaisiin määrittää alemman tason tehtäviä. Ylätason tehtävässä olisi nappi, jolla alagraafit saa piiloon ja esiin.

6.1.3 Kartta

Työkalun prototyyppiversiossa huoneen tarkassa näkymässä ei näy, mitä tehtäviä huoneessa tapahtuu ja mitä tavaroita siellä on. Huoneet ovat kovin irrallisia muusta suunnitelmasta. Työkalun ensimmäisessä versiossa kannattaakin näyttää, mitä tavaroita huoneessa on ja mitä tehtäviä siellä tapahtuu, ja toisinpäin.

Kartalle voitaisiin määrittää alueita. Esimerkiksi ensimmäisessä luvussa voisi olla alueet kartano, kaupunki ja aarresaaren viereinen saari. Vaikka niillä ei toiminnallisesti olisikaan merkitystä, kokonaisuuden jakaminen pienempiin osiin kuin koko kartta auttaisi hahmottamisessa. Toisaalta koska huone voidaan asettaa mihin kohtaan tahansa näkymää, sommittelulla voidaan ilmentää eri kokonaisuuksia, kuten testipelin kartassa kuvassa 6.4 on tehty. Alueiden nimeämismahdollisuus voisi kuitenkin selventää karttaa muillekin kuin suunnittelijalle.



Kuva 6.4: Testipelin kartta työkalussa.

Työkalu tarjoaa liittämismahdollisuuden jokaiselle huoneen reunalle, mitä voidaan käyttää kuvastamaan eri ilmansuuntia, joihin pelissä voidaan mennä. Huoneet tulee luonnollisesti liitettyä niin, kun huoneet ovat selvästi vierekkäisiä maailmassa. Jos taas huoneesta mennään esimerkiksi autolla kauemmas, työkalussa ei välttämättä voida liittää huoneita sivuilta, joilta pelaaja toiseen huoneeseen lähtisi. Esimerkkipelissä kartanon huoneiden yhteydet eri sivuilta kuvaavat yhteyksien suuntia niin kuin ne tulevat olemaan pelissäkin. Liitteessä E on tarkempi kuva kartasta. Kuitenkin siirtymät tapahtuvat ovista eivätkä koko reunasta. Downtown-huoneen yhteydet sivuille ja ylös kuvaavat oikeita yhteyksiä koko reunalta. Downtown-huoneesta pääsee ovista useisiin taloihin, joten näitä yhteyksiä ei pystytä kuvaamaan työkalussa siten kuin ne tulevat olemaan pelissä. Sen sijaan niiden merkitsemiseen on käytetty alareunan vapaana olevaa yhteyttä. Työkaluun ei pystytä merkitsemään, mistä yksittäisestä kohtaa huonetta pääsee mihinkin. Suunnittelijan pitää halutessaan kommunikoida tämä tiimilleen muilla tavoin.

Esimerkkipelissä pelaaja voi keskustella aluksi kartanon etupihalla olevan kuljettajan kanssa ja valita, mihin paikkaan kaupungissa haluaa mennä, kirjastoon vai lentokentälle, tai niistä takaisin kartanolle. Vaihtoehtoja on vain kolme, mutta jos niitä olisi enemmän, valinta olisi todennäköisesti pelissä toteutettu maailmankarttanäkymänä, josta valitaan haluttu paikka. Työkalussa voidaan piirtää kaikille vaihtoehdoille yhteydet toisiinsa, mutta kun huoneiden määrä kasvaa, karttagraafista tulee nopeasti sekava ja yhteyksiä unohtuu. Aiemmin ehdotettiin alueiden määrittämistä, mikä olisi tässäkin tilanteessa hyödyllinen. Joko alueelle voitaisiin merkitä esimerkiksi valintalaatikko, että sieltä on vapaa kulku joka paikkaan, tai ainakin alueen ja sen nimen avulla esittää tieto muualla. Yksi vaihtoehto olisi, että karttaan voidaan lisätä kommentteja. Kommenttien avulla voitaisiin myös merkitä muistiin ehtoja, jotka rajoittavat huoneisiin pääsyä.

Kun on suunniteltu, miten huoneet ovat suhteessa toisiinsa, se auttaa huoneiden piirtämisessä. Työkalussa ei voida piirtää kuvia. Kätevää olisi, että kuvan saisi suoraan työkalun kautta auki toiseen, käyttäjän valitsemaan kuvankäsittelyohjelmaan. Se olisi kuitenkin jo hienosäätöä; muualla tehdyn tallennetun kuvan liittäminen työkaluun on helppoa.

Kartta kuvaa yhden luvun huoneet. Karttaan voitaisiin merkitä, mistä huoneesta luku alkaa ja mihin se loppuu, eri aloituksilla ja lopetuksilla. Tämä tieto käy vähintään implisiittisesti ilmi tehtävistä. Se voisi kuitenkin selventää karttaa ja tarinan kulkua muille lukijoille.

Välianimaatiot voivat tapahtua huoneissa, joihin pelaaja ei voi mennä muuten pelissä. Tällaisetkin huoneet ja yhteydet voidaan piirtää karttaan. Työkalu ei generoi automaattisesti pelirunkoon siirtymisiä toisiin huoneisiin.

6.1.4 Suunnitelman jäsentely

Työkalu voi auttaa etenkin tuoretta suunnittelijaa; se näyttää ja jäsentelee asioita, joita seikkailupeliä suunnitellessa tulee tehdä. Kokenut suunnittelija saattaa haluta suunnitella pelin tavalla, johon hän on tottunut ja jonka hän on hyväksi todennut. Jos työkalu ei taivu siihen, se saattaa jäädä käyttämättä. Työkalu on luotu menetelmien pohjalta, joita alalla käytetään. Se tarjoaa toimivan tavan suunnitella seikkailupelejä.

Työkalu helpotti ideoiden syntymistä. Tosin se johtui ainakin osittain siitä, että työkalun tekoa varten opiskeltiin, miten seikkailupelejä kannattaa suunnitella. Joka tapauksessa työkalu tukee menetelmien käyttöä hyvin. Työkalun ansiosta suunnitelmat ovat muistissa siistissä järjestyksessä ja niitä on helppo muokata, toisin kuin voisi olla paperilla tai pelkässä tekstinmuokkausohjelmassa. Työkalu ei kuitenkaan rajoita luovuutta, vaan tehtäville on helppo keksiä lisää tasoja ja taustatarinaa. Työkalu ei rajoita millaisia tehtävät voivat olla, koska ne kirjoitetaan loppujen lopuksi vain tekstinä. Se järjestelee ja antaa niille yhtenäisen rakenteen. Huonona puolena on, että työkaluun ei voida liittää luonnoskuvia vapaasti, kuten paperilla voitaisiin. Esimerkiksi tehtäviin ei voida liittää lainkaan kuvia. Jotkut asiat on helpompi esittää kuvana kuin tekstinä. Esimerkiksi jonkin laitteen toiminta saattaisi olla helpompi esittää kuvasarjana kuin tekstinä. Tehtävien kuvien lisäämisen mahdollisuutta harkitaan jatkokehitys-vaiheessa.

6.1.5 Tavaroiden uudelleenkäytettävyys tehtävissä

Saatavilla olevien tavaroiden näyttäminen tehtävässä voi auttaa tavaroiden uudelleenkäyttämisessä, etenkin jos siihen pyritään tietoisesti: Käydään saatavilla olevien tavaroiden lista läpi ja mietitään jokaisen tavaran kohdalla, voitaisiinko sitä hyödyntää tehtävän ratkaisemiseen. Se auttaa keksimään tavaralle käyttötarkoituksia, joita ei heti tulisi mieleen. Ratkaisuja kannattaa pohtia myös saatavilla olevien tavaroiden ulkopuolelta, sillä tavaroiden uudelleenkäyttö ei ole aina paras ratkaisu.

Ominaisuuksien lisääminen tavaralle jää helposti tekemättä, kun sitä ei voida tehdä samassa näkymässä tehtävien suunnittelun kanssa. Vasta kun testipelin ensimmäisen luvun tehtävät oli suunniteltu, tarkasteltiin tavaralistauksesta, mitä tavaroita oli luotu. Tällöin voidaan helposti myös lisätä ominaisuuksia, mutta niistä ei varsinaisesti ole enää hyötyä ensimmäisen luvun suunnittelussa, ellei haluta palata muokkaamaan tehtäviä tai tekemään vielä lisää tasoja tehtäviin. Jos tavarat ovat käytettävissä vielä seuraavassakin luvussa, silloin ominaisuuksien lisäämisestä voi olla vielä hyötyä.

6.1.6 XML:n luonti

Testipelistä generoitiin pelimoottoreihin vientiä varten XML-tiedosto, joka on esitetty liitteessä C. Seuraavaksi seikkailupelitiimien tulisi testata, kuinka monimutkaista on

luoda komponentti, joka vie datan heidän käyttämäänsä pelimoottoriin. Alustavasti tutkittiin työkalun datan viemistä Visionaireen.

Kohdassa 4.4 kuvatuissa pelimoottoreissa ja niiden XML-tiedostoissa on kahdenlaisia tavaroita: tavaralistauksen tavarat sekä huoneissa olevat tavarat. Jos tavara esiintyy sekä huoneessa että tavaraluettelossa, sille on kaksi objektia, vaikka ne kuvastavat samaa tavaraa. Tavara myös näyttää seikkailupeleissä usein erilaiselta tavaraluettelossa kuin huoneessa. Kaikki tavarat työkalun prototyypissä ovat kyseisten pelimoottorien kannalta tavaraluettelon tavaroita, sillä niille ei ole määritelty sijaintia tietyssä huoneessa. Visionaire kaatuu, jos tavaralistauksen tavara yritetään kopioida ja liittää huoneeseen Visionairen copy-paste-napeilla. Vaikuttaa siltä, että tavara ei voi olla huoneessa, jos sen *ObjectIsItem*-arvo on tosi. Tavarat ovat joko *Items*-näkymän tavaroita, tai huoneen tavaroita *Scene objects* -näkymässä.

Työkalun prototyyppiversio tukee vain yhdenlaisia tavaroita. Koska niille ei ole määritelty huonetta, jossa ne sijaitsevat, ne ovat Visionairen kannalta tavaraluettelon tavaroita. Työkalussa tavara voi kuitenkin olla sama molemmantyyppisille objekteille. Työkalu voi generoida vasta XML-tiedostoon erilliset objektit. Koska molemmat objektit edustavat samaa tavaraa, on selkeää, että työkalussa on vain yksi tavara. Tavaralle tulee määritellä työkalussa, mihin huoneeseen tai huoneisiin tavara kuuluu. Tämä voidaan määrittää joko suoraan tavaran ja huoneen välille tai tehtävän kautta. Jos tehtävälle voidaan lisätä tapahtumapaikaksi huone, tehtävässä esiintyvät tavarat linkittyvät huoneeseen. Tavaran lisänäkymässä tavaralle tulisi voida liittää kaksi kuvaa: kuva tavaraluettelossa sekä huoneessa.

Saattaa olla monimutkaista luoda koko XML-tiedosto pelimoottoria varten etukäteen, niin ettei virheitä tule, koska siinä on paljon tunnisteita ja linkkejä. Kun kopioidaan Visionairen luoma XML-tiedosto toiseen kansioon, peli toimii suoraan; siinä ei ole viitteitä muihin tiedostoihin, joten pelkän XML-luomisen pitäisi riittää projektin luomiseen.

Toinen vaihtoehto on luoda tyhjä projekti ja antaa Visionairen tehdä XML-tiedosto pohjaksi, jota sitten muutetaan. Mahdollisuus XML-tiedoston muokkaamiseen olisi joka tapauksessa tehtävä, jotta rungon generointi voidaan tehdä suunnitelman muutosten jälkeenkin eikä vain kertaluontoisesti. Jos pelimoottorissa on jo määritelty lisää tietoa XML-tiedostoon, se ei häviä, kun muutetaan elementtejä eikä luoda koko XML-tiedostoa uudelleen. Työkalussa on määritelty tunnisteet tehtäville. Työkalun pitäisi määrittää yksikäsitteiset tunnisteet kaikille objekteille, jotta ne löydetään pelimoottorin XML-tiedostosta. Tällöin työkalussa voidaan tehdä myös samannimisiä objekteja.

Pelin juoni ja tehtävät luodaan AGS:ssä ja Visionaireissa skriptaamalla. Skriptien generointi olisi monimutkaista ja ohjelmoija joutuisi todennäköisesti kuitenkin muokkaamaan niitä. Automaattisen skriptien luonnin sijaan objekteihin liittyviin

skripteihin voidaan lisätä kommentit, mitä niissä pitäisi tehdä. Kommenttien avulla ohjelmoija näkisi helposti, mitä skriptin tulisi tehdä, mutta voisi toteuttaa sen alusta alkaen itse. Tehtävien ongelman ja ratkaisun kuvaukset, voitaisiin kirjoittaa niihin liittyviin tavaroihin tai paikkoihin. Lukujen kuvaus taas on niin yleisellä tasolla, että se ei luontevasti liity mihinkään objektiin, vaan sen voisi liittää erilliseen tyhjään skriptitiedostoon, tai se olisi katseltavissa vain työkalusta.

6.2 Uudet ominaisuudet

Seuraavassa on esitetty uusia ominaisuuksia työkaluun. Työkalun sujuvaa käyttöä helpottaisivat myös pienet perusominaisuudet, kuten luvun huoneiden lisääminen toisiin lukuihin. Niitä ei ole tässä pohdittu.

6.2.1 Pelin kulku

Tehtävissä on mahdollista, että ratkaisu löytyy ennen ongelmaa. Olisi järkevää saada ensin tehtävänanto eli tavoite ennen kuin pelaaja voi esimerkiksi kerätä tavaran, jolla tehtävä ratkaistaan [14]. Muuten pelaaja kerää kaikki tavarat, jotka voidaan ottaa mukaan, tietämättä, miksi hän niitä mukanaan kantaa [14]. Seikkailupeleissä pelaaja olettaa, että jos tavara voidaan ottaa, sitä myös jossain kohtaa tarvitaan [14]. Ongelma ennen ratkaisua on välillä vaikea toteuttaa, sillä ratkaisunkin on kuitenkin oltava saatavilla [14]. Dialogeihin saa lisää vaihtoehtoja usein vasta kun on nähnyt tietyn asian. *Silence: The Whispered World 2* -pelissä näyttää ennakkoon julkaistujen pelivideoiden perusteella olevan tehtäviä, joissa tavaroita pääsee käyttämään vasta kun tehtävä on selvillä [82] [83]. Pelissä ei näytä myöskään olevan tavaraluetteloa, vaan tehtävät suoritetaan pienillä alueilla [82] [83].

Työkalussa käytetään tehtävien esittämiseen tehtävägraafia, joka pohjaa suoraan Puzzle Dependency Chart -kaavioon. Se kuvaa siis riippuvuuksia, eikä pelin kulkua. Sen takia siinä ei voida suunnitella, että tehtävänanto esitetään pelaajalle ennen ratkaisua ja järjestystä toimintojen suorittamiselle. Kartan avulla suunnittelija voi tarkastaa, että huoneeseen, jossa on ratkaisu, ei pääse kuin huoneen, jossa ongelma esitetään, kautta. Tästä tulisi kuitenkin nopeasti monimutkaista, kun ratkaisu-ongelma-pareja lisätään. Lisäksi tämä saattaisi johtaa putkimaisiin karttoihin.

Suunnittelija ei voi olettaa, että pelaaja suorittaa asiat tietyssä järjestyksessä, jos toisenlaisetkin suoritusjärjestykset ovat mahdollisia. Tarkastellaan esimerkkinä seuraavaa tilannetta: Pelissä on hotelli, jonka edestä pelaaja aloittaa. Suunnittelija olettaa, että pelaaja menee heti sisään hotellin etuovista. Tällöin vastaanottovirkailija tervehtii pelaajahahmoa ystävällisesti ja kertoo, että hotelli on valitettavasti täynnä. Hotellin edestä pääsee kuitenkin kävelemään katua pitkin myös muualle. Pelaaja ei menekään heti hotelliin sisälle sen edustalta, vaan kulkee katuja pitkin hotellin takaovelle ja pyrkii sieltä sisään. Siellä hän tapaa vastaanottovirkailijan ensimmäistä kertaa ja vastaanotto-

virkailija sanoo vihaisena: ”Kuten jo kerroin, hotelli on täynnä!”. Suunnittelija ei ole ottanut huomioon, että pelaaja voi mennä takaovelle ennen kuin hän käy vastaanottotiskillä. Tällöin vastaanottovirkailija ei ole kertonut pelaajalle, että hotelli on jo täynnä, vaikka hän näin väittääkin. [47]

Esimerkin virhe voi hämmentää tai ärsyttää pelaaja, mutta ei vaikuta pelin kulkuun. Toisenlainen oletus suoritusjärjestyksestä voi kuitenkin johtaa tätä kriittisempiinkin ongelmiin, kuten umpikujaan pelissä. Pelaaja voi suorittaa toimintoja niin monessa järjestyksessä, että suunnittelijan ei ole tarkoituksenmukaista kirjata niitä kaikkia ylös. Sen sijaan, esimerkiksi tilanteessa, joka on kuvattu yllä, NPC-hahmon dialogiin tulisi liittää muuttuja, joka määrittää, onko pelaaja vielä puhunut hahmolle.

6.2.2 Umpikujien ehkäisy

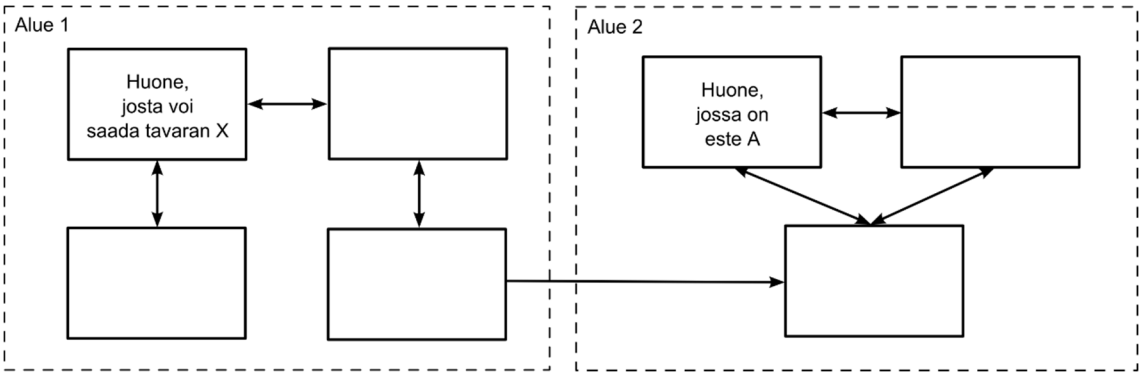
Seikkailupeleissä ei pidetä hyvänä asiana, jos tehtävä voidaan ratkaista väärin tai muulla tavoin joudutaan umpikujaan [41]. Joissain vanhoissa seikkailupeleissä on tilanteita, joissa voidaan joutua umpikujaan. Esimerkiksi *King's Quest 5* -pelissä on tehtävä, jossa pitää heittää kissaa saappaalla, jotta pelastaisi hiiren. Tilanne tapahtuu kuitenkin vain kerran. Jos siinä epäonnistuu, selviää vasta myöhemmin, kun päähahmo on vangittu kellariin köysiin sidottuna, että peliä ei voida enää päästä läpi. Jos hiiren on pelastanut, se tulee puremaan köyden poikki ja pelaajahahmo pääsee pakenemaan. Jos peli on jaettu lukuihin, on huomioitava, että jos pelaaja tarvitsee esimerkiksi toisessa luvussa tavaraa, jonka saa ensimmäisessä luvussa, täytyy olla jokin ehto, että pelaaja ei pääse seuraavaan lukuun, ennen kuin hänellä on kyseinen tavara [14].

Työkalun prototyyppi ei tee mitään tarkistuksia siitä, voiko peli päätyä umpikujaan. Niiden välttäminen on suunnittelijan vastuulla. Jotkut umpikujaan johtavat toiminnot ovat melko selkeitä. Esimerkiksi jos tavara on kertakäyttöinen eikä sitä saa mistään lisää, tavaraa ei saa käyttää muualle kuin tehtävään, johon se on tarkoitettu. Ideaalisesti työkalu ilmoittaisi kaikista umpikujan mahdollisuuksista.

Tavara voi olla sellainen, että pelaajalla on mahdollisuus saada se, mutta hän ei ota sitä. Näin on mahdollista, että pelaaja ei ole ottanut tavaraa, vaikka se olisi ollut mahdollista, ja hän on jo edennyt pelissä kartalla eri alueelle, josta ei pääse enää takaisin hakemaan tavaraa. Jos pelaaja tarvitsee tavaraa uudella alueella tapahtuvassa tehtävässä, hän päätyy umpikujaan, ellei toisia ratkaisuja ole. Työkalun tulisi siis huomioida, että pelaajan on välttämätöntä saada tavara ennen uudelle alueelle menoa. Pelaajalla on varmasti tavara uudella alueella, jos sitä tarvitaan vanhan alueen tehtävässä. Pelaaja voi myös saada tavaran, kun hän suorittaa tehtävän, joka välttämättä suoritetaan, ennen kuin alueelta pääsee pois.

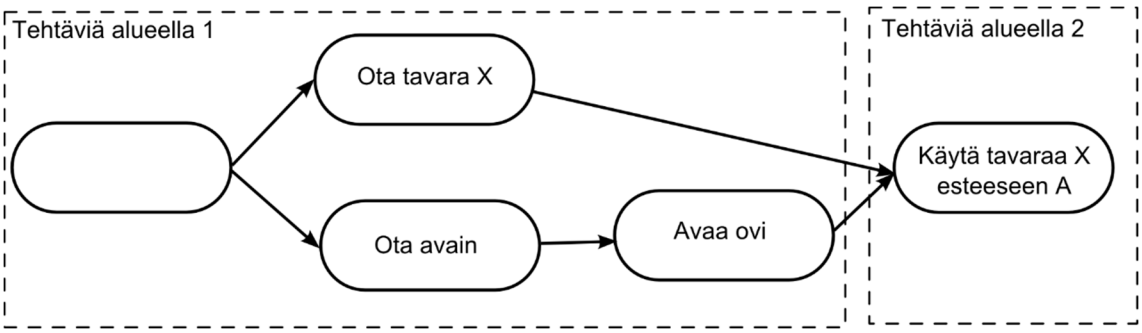
Kuvassa 6.5 on esitetty kartta. Yksisuuntainen yhteys määrittää alueet. Alueella 1 on neljä huonetta, joiden välillä pelaaja voi liikkua edestakaisin. Alueella 2 on kolme

huonetta, joiden välillä voidaan liikkua vapaasti. Kun pelaaja siirtyy alueelle 2, hän ei pääse enää halutessaan takaisin alueelle 1.



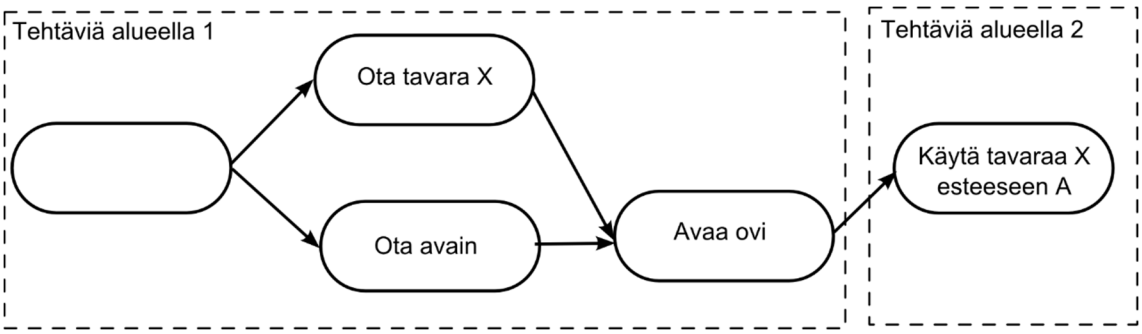
Kuva 6.5: Kartta. Suorakulmiot edustavat huoneita. Huoneet, joiden välillä pääsee liikkumaan edestakaisin, muodostavat alueen.

Alueella 2 on este, jonka ratkaisuun tarvitaan tavaraa X, joka on saatavilla alueella 1. Kuvassa 6.6 on esitetty tehtävägraafi. Alueelta 1 pääsee eteenpäin, kun ratkaisee tehtävän ”Avaa ovi”. Tehtävä ”Ota tavara X” ei ole riippuvainen tehtävästä, jonka kautta alueelta pääsee pois, joten pelaaja ei ole välttämättä suorittanut tehtävää ja saanut tavaraa X. Koska pelaaja tarvitsee tavaraa alueen 2 tehtävän ratkaisemiseen, peli voi päätyä umpikujaan.



Kuva 6.6: Tehtävägraafi. Alueelle 2 tulee useampi kuin yksi riippuvuus alueen 1 tehtävistä. Tällöin peli voi päätyä umpikujaan, jos pelaaja on poistunut alueelta 1 ennen kuin kaikki riippuvat tehtävät on suoritettu.

Kuvassa 6.7 on esitetty tehtävägraafi, jossa ongelmaa ei tapahdu. Tehtävä ”Ota tavara X” on riippuvainen tehtävästä, jonka kautta alueelta pääsee pois, joten pelaajalla on varmasti tavara X ennen kuin hän poistuu alueelta. ”Ota tavara X” voisi yhtä lailla olla esimerkiksi tehtävää ”Ota avain” edeltävä tehtävä, kunhan se on lopulta riippuvainen tehtävästä ”Avaa ovi”. Yksi vaihtoehto olisi, että tavaran saa pakolla, kun tehtävä ”Avaa ovi” on suoritettu. Ennen kuin alueelta siirrytään toiseen, tulee syntyä pullonkaula-tehtävä. Tämä on nähtävissä myös testipelin tehtävägraafista liitteessä D.



Kuva 6.7: Tehtävägraafi. Alueiden 1 ja 2 välillä on vain yksi riippuvuus. Kaikki alueen 1 tehtävät ovat riippuvaisia tehtävästä, jonka kautta alueelta pääsee seuraavalle alueelle.

Tehtäviin tulee merkitä, missä huoneissa ne tapahtuvat, jotta työkalu voi kartan yhteyksien avulla määrittää, mihin alueeseen ne kuuluvat. Tällöin on mahdollista tarkastella umpikujan mahdollisuutta: Jos eri alueiden tehtävien välillä useampi kuin yksi riippuvuus, umpikuja voi syntyä. Kun työkalu huomaa umpikujan mahdollisuuden, se voisi esimerkiksi näyttää varoitustekstin, mikä on vikana, sekä värittää osalliset tehtävät huomiovärillä. Suunnitelmaan voi muodostua tilanteita, joissa alueen määrittäminen on monimutkaisempaa kuin edellä kuvatussa mallissa. Erilaisten tilanteiden löytäminen ja kattaminen vaatisi lisätutkimusta ja lisäominaisuuksia työkaluun. Esimerkiksi pelissä voi olla huone, johon pelaajahahmo lukitaan, mutta tehtävän ratkaistuaan pelaaja pääsee taas liikkumaan vapaasti edestakaisin koko alueella. Tällaista tilannetta varten kartan yhteyksiin tulisi voida lisätä ehtoja.

Plotkin [84] on kehittänyt skriptiä, joka tarkastaa, että tehtävien kulku on toimiva ja siihen ei synny huomaamattomia umpikujia. Työkalu on tekstipohjainen. Jokainen toiminto ja sen ennakkoehdot ja seuraukset määritellään syötteessä. Esimerkiksi toiminto ”mene luolaan” vaatii tavaran ”lamppu”. Toiminnon seurauksena pelaaja on luolassa. Syöterivi on esitetty seuraavassa:

```
EnterCave = Chain(Has(lamp=True), Set(underground=True))
```

Työkalu tulostaa mahdolliset lopputilat. Lopputila kuvaa, mitä tavaroita pelaajalla on tai mitä ehtoja on voimassa, sekä minkä toimintojen kautta tilaan on päästy. Alla on esimerkki lopputilasta:

```
<lamp pants sword underground>
  (4): FindLamp, EnterCave, FindSword, FeedOrc
```

Työkalulla voidaan myös piirtää graafi tiloista Graphvizin avulla. Plotkin toteaa, että työkalu ylikuormittuu helposti, koska tiloja tulee paljon. [84]

Suunnittelutyökalussa olisi hyvä vastaavanlaisesti tarkistaa, että pelissä on mahdollista päästä vain yhteen, haluttuun lopputilaan.

6.2.3 Vaihtoehtoiset ratkaisut tehtäviin

Tehtävät, joihin on vain yksi ratkaisu, eivät ole enää hauskoja, kun pelaaja on ne kerran ratkaissut [14, s. 446]. Jos tehtävään on monta ratkaisua, se lisää pelin uudelleen-pelattavuutta. Jos tehtävä voidaan ratkaista usealla tavalla, sen olisi hyvä olla mahdollista pelaajalle. Muuten hän voi turhautua, kun hänelle täysin looginen ratkaisu ei kelpaakaan. Esimerkiksi tehtävänä on katkaista naru. Pelaajalla on useampi terävä tavara tavaraluettelossaan, joten kaikkien pitäisi käydä tehtävän ratkaisemiseen, ellei suunnittelija selitä pelaajalle, miksi vain tietty tavara käy. Myös näin saadaan lisää pelattavaa, mikäli pelaaja on kiinnostunut etsimään vaihtoehtoisia ratkaisuja. Toisaalta jos peli on hintansa arvoinen ensimmäisellä pelikerralla, ei sen tarvitsekaan olla uudelleen pelattavissa [14].

Jos tehtäviin on vaihtoehtoisia ratkaisuja, tehtävien riippuvuudet voivat olla joko vaihtoehtoisia tai pakollisia. Esimerkiksi tehtävän ”Avaa ovi”, jonka ongelma on, että ovi on lukossa ja saranat ovat niin ruosteessa, että ovi ei toimi, edellä olevat riippuvat tehtävät voivat olla ”Hanki avain” ja ”Hanki öljyä”. Nämä molemmat pitää suorittaa, jotta oven saa auki. Kuitenkin vaihtoehtoisena ratkaisuna voisi olla oven räjäyttäminen, jota varten ennen tehtävää olisi riippuvuus ”Hanki räjähde”. Jotta vaihtoehtoiset ja keskenään pakolliset riippuvuudet voidaan erotella, tehtävällä voisi olla useampi sisääntulo, joihin voidaan edelleen liittää useita riippuvuuksia. Vaihtoehtoiset tehtävät liitettäisiin eri sisääntuloihin.

Ominaisuuksien lisääminen tavaroille voi helpottaa suunnittelijaa löytämään useita vaihtoehtoisia ratkaisuja. Pidemmälle vietynä suunnittelija voisi laittaa vaihtoehtoisiksi ratkaisuksi esimerkiksi veden kantamiseen kaikki tavarat, joilla on ominaisuus ”can carry liquid”, eikä tiettyjä tavaroita. Tämä voisi luoda pelistä perinteisiä pelejä paljon vapaamman, mutta se lisäisi pelin monimutkaisuutta suunnittelijan näkökulmasta ja tällöin umpikujia voisi syntyä tavallista helpommin. Umpikujien tarkastaminen ohjelmallisesti olisi siis erityisesti tällöin tarpeellista.

6.2.4 Hahmot, tieto ja tavarat objekteina

Grim Fandango -pelin suunnitteludokumentissa hahmoista on esitetty nimi ja rivin mittainen kuvaus, joka kertoo esimerkiksi hahmon ammatin tai miksi hahmo on pelissä [42]. Työkalun ensimmäisessä versiossa ei ole toteutettu lainkaan hahmoja. Ainoa hahmoille tyypillinen ominaisuus on, että niiden kanssa voidaan keskustella. Hahmot voidaan kuitenkin luoda tehtäviin hotspot-kohteina, eli tavaroina, joita pelaaja ei voi ottaa, koska loppujen lopuksi ne eivät muuta ole. Hahmo voi tosin joskus olla myös tavara: Esimerkiksi *The Curse of Monkey Island* -pelissä on pääkallo, jonka kanssa voidaan keskustella, mutta se myös otetaan yhdessä vaiheessa tavaraluetteloon. On siis hyväkin, ettei työkalu liikaa rajoittaisi objekteja hahmoiksi tai tavaroiksi. Tavara ja hahmot voisivatkin olla vain peliobjekteja, josta tavara on prototyypissä periytetty.

Samaan tapaan pelaajahahmon hankkima tieto voisi olla peliobjekti, ja työkalun ensimmäisessä versiossa se voitaisiin siis esittää tavarana. Tieto voidaan saada samalla lailla kuin tavara. Nykyisessä tehtävän Available items -listassa näkisi siis myös, mitä pelaajahahmo tietää kyseisen tehtävän kohdalla.

Objektin ominaisuuksiin voidaan haluttaessa liittää myös metadataa, onko peliobjekti tavara, hahmo, tietoa vai useampi näistä. Tällöin ominaisuutta hakemalla pystyttäisiin tarkastelemaan esimerkiksi vain hahmoja objektilistauksessa.

6.2.5 Dialogit

Dialogi voisi olla liitettävissä mihin tahansa objektiin. Dialogin sujuvaa luomista varten tulisi kehittää dialogieditori. Dialogieditori on niin iso kokonaisuus, että sitä ei toteutettu diplomityön puitteissa. Sellaisen lisääminen voisi kuitenkin olla hyödyllistä, sillä AGS:n ja Visionairen dialogieditoreista ei näe kovin hyvin dialogin kulkua. AGS:ssä dialogia tehdään lomakkeena ja dialogiskriptinä. Hyvänä puolena on, että dialogiskriptiin voidaan lisätä myös toiminnallisuutta: Esimerkiksi pelaajahahmo kääntyy katsomaan jotain kohdetta, josta puhutaan. Visionairessa dialogeilla on selkeämpi rakenne kuin AGS:ssä, mutta senkään esitys ei ole kovin visuaalinen. Työkalussa dialogieditori voitaisiin esittää graafina. Sen toteuttamista tulisi kuitenkin tutkia lisää.

6.2.6 Komennot

Työkalussa voitaisiin määritellä komentokuvakkeet, kuten käsi, suu ja silmä. Niihin voitaisiin liittää verbejä, joihin voidaan viitata tehtävien ratkaisu-kentässä. Tehtävän ratkaisu-kentässä voisi olla syntaksi, jolla lisätään verbejä, samaan tapaan kuin tavaroita lisätään. Verbejä voitaisiin sitten liittää sopivaan komentokuvakkeeseen erillisessä näkymässä. Kuvakkeelle voitaisiin määrittää nimi, kuvaus ja tiedostopolku sen kuvaan.

6.2.7 Kehitysprosessi

Työkalu on tarkoitettu nykyisellään seikkailupelin suunnitteluun suunnittelijoiden toimesta sekä suunnitelman dokumentoimiseen ja jakamiseen tiimin kesken. AGS:n keskustelupalstalla on ideoitu samantyylistä työkalua [85]. Keskustelussa ehdotetaan, että työkalulla voitaisiin myös suunnitella ja seurata projektin kulkua [85]. Työkaluun voitaisiin määritellä työtehtäviä, joita tiimin jäsenet ottavat itselleen [85]. Työtehtäviin voitaisiin myös arvioida, kuinka paljon niiden tekemiseen menee aikaa, ja priorisoida niitä [85].

Tiimin eri rooleissa oleville käyttäjille voisi olla erilaisia näkymiä: Graafikot näkisivät listauksen, mitä huoneita ja tavaroita tulee piirtää ja mitä on jo tehty. Ohjelmoijat taas näkisivät, mitkä tehtävät on toteuttamatta. Myös testaajille voisi olla näkymä ja

määritellä työtehtäviä, mitä tulee testata. Testaajat voisivat sitten kirjoittaa bugeja ja kommentteja ylös liittyen eri elementteihin ja kohtiin pelissä. Bugeista voitaisiin muodostaa suoraan tehtäviä ohjelmoijille.

Yllä esiteltyjen ominaisuuksien toteuttaminen vaatisi pitkäaikaista kehitystyötä. Työkalu ei olisi tarkoitettu enää vain pelin suunnitteluun vaan myös projektin suunnitteluun. Pitäisikin tutkia tarkemmin, toisivatko tällaiset ominaisuudet hyötyä osana työkalua. Projektin suunnitteluun ja seuraamiseen on jo olemassa monenlaisia työkaluja [86], joten niiden lisääminen työkaluun saattaa olla tarpeetonta. Ominaisuuksien lisääminen myös vaikeuttaa työkalun käytön oppimista.

7. YHTEENVETO

Työssä tutkittiin minkälainen työkalu ja ominaisuudet ovat hyödyllisiä seikkailupelin suunnitteluvaiheessa. Työhön saatiin kerättyä menetelmiä, työkaluja ja ohjeita, joiden avulla seikkailupelejä kannattaa suunnitella. Näitä hyödyntäen määriteltiin suunnittelutyökalun ominaisuudet. Työkalusta toteutettiin prototyyppi. Työkalun ominaisuuksia testattiin suunnittelemalla testipeli työkalussa. Työkalu palveli tarkoitustaan hyvin, mutta vaatii jatkokehitystä, koska kaikkia välttämättömiä ominaisuuksia ei ole toteutettu tai viimeistelty.

Toisena kysymyksenä tutkittiin, miten suunnitteluvaiheessa tehty työ saataisiin hyödynnettyä seikkailupelin toteutuksen pohjan luomisessa. Työkalu luo geneerisen XML-tiedoston. Pelinkehittäjät voivat luoda komponentin, joka lukee tiedon heidän käyttämäänsä pelimoottoriin. Tämä vaatii aluksi työtä, mutta on sen jälkeen uudelleenkäytettävissä seuraavissa projekteissa. Tutkittiin myös alustavasti, kuinka mutkatonta XML:n tuonti pelimoottorin muotoon käytännössä olisi. Tässä huomattiin joitakin ongelmia, jotka vaatisivat työkaluun lisää ominaisuuksia. Seuraavaksi kannattaakin kokeilla XML:n generointia laajalti käytettyihin pelimoottoreihin kokonaisuudessaan, jotta saadaan selville, mitä muita ongelmia siinä voi esiintyä.

Työkalu helpotti etenkin tehtävien suunnittelua. Tosin se johtui ainakin osittain siitä, että työkalun tekoa varten opiskeltiin, miten seikkailupelejä kannattaa suunnitella. Työkalu siis mahdollistaa suunnittelun oppien mukaan, vaikka ei välttämättä ohjaa siihen. Työkalua arvioimaan tarvittaisiin ulkopuolisia, sekä uusia että kokeneita seikkailupelisuunnittelijoita ja tiimejä. Seuraava vaihe olisi toteuttaa prototyypin kriittisimmät osat valmiiksi, kuten projektin tallentaminen ja lataaminen, jotta työkalua on helpompi testata. Tämän jälkeen kannattaa pyytää seikkailupelisuunnittelijoita testaamaan työkalun prototyyppiä ja antamaan palautetta siitä. Tällöin saadaan selville, onko työkalu menossa oikeaan suuntaan ja mitä ongelmia siinä mahdollisesti on. Lisäksi voidaan tiedustella, mitä ominaisuuksia työkaluun kannattaa seuraavaksi toteuttaa. Myös työssä esitettyjä jatkokehitysajatuksia voidaan priorisoida palautteen mukaan. Ennen ensimmäistä julkaistavaa versiota työkalun käytettävyyttä ja ulkonäköä kannattaa parantaa, sillä niihin ei ole diplomityön puitteissa kiinnitetty huomiota.

Työkalun prototyyppiversio voidaan tarjota ladattavaksi esimerkiksi AGS:n keskustelupalstan kautta. Siellä voidaan aloittaa keskustelu, jossa toivotaan palautetta työkalusta. Myöhemmästä julkaisukelpoisesta versiosta harkitaan, tarjotaanko sitä käyttäjille kaupallisena vai ilmaisohjelmana.

LÄHTEET

[1] M. J. P. Wolf, The Video Game Explosion, A History from PONG to PlayStation and Beyond, Westport Connecticut, London: Greenwood Press, 2008, p. 81–90.

[2] ”Sierra Entertainment, Inc.,” MobyGames, [Online]. Saatavissa: http://www.mobygames.com/company/sierra-entertainment-inc. [Haettu 14.6.2015].

[3] R. DeMaria and J. I. Wilson, High Score! The Illustrated History of Electronic Games, 2nd ed., Emeryville California: McGraw-Hill/Osborne, Brandon A. Nordin, 2004.

[4] K. Williams, ”Introduction to The Roberta Williams Anthology,” The Sierra Help Pages, [Online]. Saatavissa: http://www.sierrahelp.com/Misc/IntroductionToRWAnth.html. [Haettu 1.3.2015].

[5] W. Hunter, ”Mystery House - Apple II (Sierra Online 1980),” YouTube, 15.1.2008. [Online]. Saatavissa: https://youtu.be/NrH4AJ_q7FA. [Haettu 1.3.2015].

[6] M. Barton ja B. Loguidice, ”A History of Gaming Platforms: The Apple II,” UBM Tech, Gamasutra, 31.1.2008. [Online]. Saatavissa: http://www.gamasutra.com/view/feature/3527/a_history_of_gaming_platforms_the_.php?print=1. [Haettu 30.6.2015].

[7] MondoPest, ”KING’S QUEST GAME AWARDS 2014 REVEAL TRAILER,” GamingShogun.com, 8.12.2014. [Online]. Saatavissa: http://gamingshogun.com/2014/12/08/kings-quest-game-awards-2014-reveal-trailer/. [Haettu 30.6.2015].

[8] T. Wilde, ”Ron Gilbert on Disney: "It should be me that owns Monkey Island",” PCGamer, 30.11.2012. [Online]. Saatavissa: http://www.pcgamer.com/ron-gilbert-disney-monkey-island/. [Haettu 14.6.2015].

[9] R. Gilbert, ”Thimbleweed Park Development Blog, Story Layout,” Terrible Toybox, Inc., 12.1.2015. [Online]. Saatavissa: http://blog.thimbleweedpark.com/story_layout. [Haettu 14.6.2015].

[10] jsnlomberg, ”Jedi, fedoras, and carpal tunnel: A history of LucasArts,” VentureBeat, 2.11.2012. [Online]. Saatavissa: http://venturebeat.com/2012/11/02/a-history-of-lucasarts/. [Haettu 30.6.2015].

[11] Gameragi Games, ”Let's Play Myst: Part 3 -Dark Secrets, +Part 1,2,” YouTube, 1.6.2012. [Online]. Saatavissa: https://www.youtube.com/watch?v=JXA9SP5becM. [Haettu 5.3.2015].

[12] K. Andreadis, ”Revisiting Myst: One of the most successful adventure games ever,” IGN Games, 27.4.2014. [Online]. Saatavissa: http://www.ign.com/articles/2014/04/28/revisiting-myst-one-of-the-most-successful-adventure-games-ever. [Haettu 5.3.2015].

[13] PastPlayer, ”Look in Hole: Gameplay Parody in Sierras Space Quest,” Play the Past, [Online]. Saatavissa: http://www.playthepast.org/?p=3834. [Haettu 30.6.2015].

[14] A. Rollings and E. Adams, Andrew Rollings and Ernest Adams on Game Design, Berkeley California: New Riders Publishing, 2003.

[15] ”The Golden Age of Adventure Games,” Adventure Gamers, 15.2.2014. [Online]. Saatavissa: http://www.adventuregamers.com/forums/viewthread/3129/. [Haettu 14.6.2015].

[16] A. Chmielarz, ”Seven Deadly Sins of Adventure Games,” The Astronauts, 30.4.2014. [Online]. Saatavissa: http://www.theastronauts.com/2014/04/seven-deadly-sins-adventure-games/. [Haettu 10.12.2014].

[17] ”100 best-selling video games of 2012 revealed,” Metro.co.uk, 14.1.2013. [Online]. Saatavissa: http://metro.co.uk/2013/01/14/100-best-selling-games-of-2012-revealed-3351774/. [Haettu 14.6.2015].

[18] ”100 best-selling video games of 2013 revealed,” Metro.co.uk, 16.1.2014. [Online]. Saatavissa: http://metro.co.uk/2014/01/16/100-best-selling-video-games-of-2013-revealed-4265929/. [Haettu 14.6.2015].

[19] ”100 best-selling video games of 2014 revealed,” Metro.co.uk, 15.1.2015. [Online]. Saatavissa: http://metro.co.uk/2015/01/15/100-best-selling-video-games-of-2014-revealed-5023605/. [Haettu 14.6.2015].

[20] C. Moriarty, ”Telltale Games' The Walking Dead sells a million,” IGN Games, 17.5.2012. [Online]. Saatavissa: http://www.ign.com/articles/2012/05/17/telltale-games-the-walking-dead-sells-a-million. [Haettu 14.6.2015].

[21] ”In Development, Now Available,” Daedalic Entertainment GmbH, [Online]. Saatavissa: http://www.daedalic.de/en/games/. [Haettu 24.5.2015].

[22] ”Double Fine Games,” Double Fine Productions, Inc., [Online]. Saatavissa: http://www.doublefine.com/games. [Haettu 16.6.2015].

[23] ”Games,” Telltale, Incorporated, [Online]. Saatavissa: https://www.telltalegames.com/games/. [Haettu 16.6.2015].

[24] ”Our Games,” Wadjet Eye Games, [Online]. Saatavissa: http://www.wadjeteyegames.com/games/. [Haettu 16.6.2015].

[25] ”How Much Does It Cost To Make A Big Video Game,” Kotaku, 15.1.2014. [Online]. Saatavissa: http://kotaku.com/how-much-does-it-cost-to-make-a-big-video-game-1501413649. [Haettu 15.6.2015].

[26] R. Gilbert ja G. Winnick, ”Thimbleweed Park: A New Classic Point & Click Adventure!,” Kickstarter, Inc., [Online]. Saatavissa: https://www.kickstarter.com/projects/thimbleweedpark/thimbleweed-park-a-new-classic-point-and-click-adv. [Haettu 18.6.2015].

[27] E. Gera, ”Tim Schafer on his Broken Age Kickstarter adventure: 'I'd do it again',” Guardian News and Media Limited, 28.4.2015. [Online]. Saatavissa: http://www.theguardian.com/technology/2015/apr/28/tim-schafer-broken-age-kickstarter-double-fine. [Haettu 18.6.2015].

[28] ”KickstartVentures, Crowd-funded Adventure Games and RPG's, Games,” KickstartVentures, [Online]. Saatavissa: http://www.kickstartadventure.com/home/games/. [Haettu 18.6.2015].

[29] D. Gilbert, Interviewee, *Questions related to my master's thesis about adventure game design tool.* [Haastattelu]. 11.3.2015.

[30] K. Stuart, ”Ron Gilbert on The Cave and how to make adventure games,” The Guardian, 27.6.2012. [Online]. Saatavissa: http://www.theguardian.com/technology/gamesblog/2012/jun/27/ron-gilbert-the-cave-interview. [Haettu 30.5.2015].

[31] ”Indiana Jones and The Fate of Atlantis,” GOG.com, [Online]. Saatavissa: http://www.gog.com/game/indiana_jones_and_the_fate_of_atlantis. [Haettu 7.3.2015].

[32] M. Lukkarinen, ”Silence: The Whispered World 2, Niin kaunis on hiljaisuus,” *Pelit,* nro 2, p. 20, 2015.

[33] M. Jarvis, ”Japanese visual novel Danganronpa sells 200,000 units in the West,” Newbay Media, MCV, The Market for Computer Video Games, 29.4.2015. [Online]. Saatavissa: http://www.mcvuk.com/news/read/japanese-visual-novel-danganronpa-sells-200-000-units-in-the-west/. [Haettu 2.8.2015].

[34] R. Barnholt, ”The Weird World of Japanese "Novel" Games,” 1UpGames, IGN Entertainment Games, 2009. [Online]. Saatavissa: http://www.1up.com/features/weird-world-japanese-games. [Haettu 18.6.2015].

[35] T. Ohlew, ”Text Adventures: The Story of Visual Novels in America,” USGamer, 9.6.2014. [Online]. Saatavissa: http://www.usgamer.net/articles/visual-novels-in-america. [Haettu 18.6.2015].

[36] A. Doulin, ”Building A Strong Indie Game Development Team,” Gamasutra, UBM Tech, 1.7.2010. [Online]. Saatavissa: http://www.gamasutra.com/blogs/AlistairDoulin/20100107/86323/Building_A_Strong_Indie_Game_Development_Team.php. [Haettu 24.5.2015].

[37] U. Wanitschke, ”Adventure game Silence: The Whispered World 2 coming soon to PS4,” Sony Computer Entertainment Europe, 23.1.2015. [Online]. Saatavissa: http://blog.eu.playstation.com/2015/01/23/adventure-game-silence-whispered-world-2-coming-soon-ps4-2/. [Haettu 24.5.2015].

[38] J. Schell, The Art of Game Design, A Book of Lenses, Burlington, MA: Morgan Kaufmann Publishers, 2008.

[39] K. Christensen, ”Monkey Island 3 Walkthrough,” The World of Monkey Island, [Online]. Saatavissa: http://www.worldofmi.com/gamehelp/walk/monkey3.php. [Haettu 22.2.2015].

[40] World of Longplays, ”PC Longplay [001] The Curse of Monkey Island,” YouTube, 26.1.2009. [Online]. Saatavissa: http://youtu.be/nExk_lHH6Y0. [Haettu 22.2.2015].

[41] R. Gilbert, ”Puzzle Dependency Charts,” Grumpy Gamer, 10.8.2014. [Online]. Saatavissa: http://grumpygamer.com/puzzle_dependency_charts. [Haettu 24.5.2015].

[42] T. Schafer, P. Tsacle, E. Ingerson, B. Mogilefsky and P. Chan, *Grim Fandango Puzzle Document,* LucasArts Entertainment Co., 1996, p. 72.

[43] ”Twine is an open-source tool for telling interactive, nonlinear stories.,” [Online]. Saatavissa: http://twinery.org/. [Haettu 30.5.2015].

[44] R. Ingermanson, ”The Snowflake Method For Designing A Novel,” Advanced Fiction Writing, [Online]. Saatavissa: http://www.advancedfictionwriting.com/articles/snowflake-method/. [Haettu 24.5.2015].

[45] ”OmniGraffle 6 for Mac, User Manual,” The Omni Group, 2014. [Online]. Saatavissa: http://downloads2.omnigroup.com/software/MacOSX/Manuals/omnigraffle-6-manual.pdf. [Haettu 23.6.2015].

[46] D. Marshall, ”How to Design Brillo Point and Click Adventure Game Puzzles,” Gamasutra, UBM Tech, 28.3.2014. [Online]. Saatavissa: http://www.gamasutra.com/blogs/DanMarshall/20140328/214187/How_to_Design_Brillo_Point_and_Click_Adventure_Game_Puzzles.php. [Haettu 10.12.2014].

[47] B. Bishop, ”How Not To Design An Adventure Game,” SiMPLE CodeWorks, Inc., [Online]. Saatavissa: http://www.simplecodeworks.com/RJB/advent.html How Not To Design. [Haettu 27.5.2015].

[48] E. Adams, Fundamentals of Game Design, 2nd ed., Berkeley, CA: New Riders, 2010.

[49] 123Pazu, ”Resonance Walkthrough - Part 7,” YouTube, 18.6.2012. [Online]. Saatavissa: http://youtu.be/6RKU9n72Mis?t=1m38s. [Haettu 23.6.2015].

[50] Kimberly, ”Walkthrough Resonance,” JayIsGames.com, [Online]. Saatavissa: http://jayisgames.com/review/resonance.php. [Haettu 23.6.2015].

[51] B. Jamin, ”Resonance walkthrough,” 2012. [Online]. Saatavissa: http://www.gamesover.com/Resonance/Resonance%20walkthrough.htm. [Haettu 1.7.2015].

[52] ”Scrivener,” Literature & Latte Ltd., [Online]. Saatavissa: http://www.literatureandlatte.com/scrivener.php. [Haettu 24.5.2015].

[53] D. Mappin, ”Scrivener For Adventure Management,” Gnome Stew, 26.9.2013. [Online]. Saatavissa: http://www.gnomestew.com/gming-advice/scrivener-for-adventure-management/. [Haettu 24.5.2015].

[54] Literature & Latte, ”Scrivener 2 Basics - Document Templates & Custom Icons *Mac Only*,” YouTube, 10.10.2010. [Online]. Saatavissa: https://www.youtube.com/watch?v=JzvkeobCWvo. [Haettu 24.5.2015].

[55] ”King's Quest II: Romancing the Throne/Kolyma Map,” 29.7.2009. [Online]. Saatavissa: http://strategywiki.org/wiki/King's_Quest_II:_Romancing_the_Throne/Kolyma_Map. [Haettu 30.6.2015].

[56] ”Shop Games,” Her Interactive, Inc., [Online]. Saatavissa: http://www.herinteractive.com/shop-games/all-games/. [Haettu 22.6.2015].

[57] ”Visionaire, Several brilliant References,” [Online]. Saatavissa: http://www.visionaire-studio.net/cms/visionaire-studio-2.html. [Haettu 28.5.2015].

[58] M. Williams, ”Telltale's Deal with Lionsgate Brings Gaming and TV Together Again,” USGamer, 2.2.2015. [Online]. Saatavissa: http://www.usgamer.net/articles/telltales-deal-with-lionsgate-brings-gaming-and-tv-together-again. [Haettu 28.5.2015].

[59] ”About Wadjet Eye Games,” Wadjet Eye Games, [Online]. Saatavissa: http://www.wadjeteyegames.com/about/]. [Haettu 28.5.2015].

[60] ”Adventure Game Studio, All Games List,” Adventure Game Studio, [Online]. Saatavissa: http://www.adventuregamestudio.co.uk/site/games/allgames/. [Haettu 28.5.2015].

[61] T. Curtis, ”Double Fine's Kickstarter-funded adventure game to use the Lua-based Moai platform,” UBM Tech, Gamasutra, 1.5.2012. [Online]. Saatavissa: http://gamasutra.com/view/news/169585/Double_Fines_Kickstarterfunded_adventure_game_to_use_the_Luabased_Moai_platform.php. [Haettu 28.5.2015].

[62] ”MoaiSDK,” 18.4.2012. [Online]. Saatavissa: http://getmoai.com/wiki/index.php?title=MoaiSDK. [Haettu 28.5.2015].

[63] ”What is Moai SDK?,” 1.4.2012. [Online]. Saatavissa: http://getmoai.com/wiki/index.php?title=What_is_Moai_SDK%3F. [Haettu 30.5.2015].

[64] ”Licenses Visionaire Studio 4.x,” Visionaire, [Online]. Saatavissa: http://www.visionaire-studio.net/cms/licenses-visionaire-studio-4.html. [Haettu 30.5.2015].

[65] ”AGS Licensing,” Adventure Game Studio, [Online]. Saatavissa: http://www.adventuregamestudio.co.uk/site/ags/legal/].. [Haettu 30.5.2015].

[66] ”Is the new Retro look here to stay?,” Adventure Gamers, 11.6.2012. [Online]. Saatavissa: http://www.adventuregamers.com/forums/viewthread/245/P45. [Haettu 30.5.2015].

[67] C. Jones, others, N. Hodgson ja W. Luo, *Adventure Game Studio, AGS Editor .NET (Build 3.3.2.0) v3.3.2, September 2014.*

[68] A. Hartmann, D. Stoffel, S. Scheckel ja T. Dibke, *Visionaire Adventure Game Engine, Visionaire 4.1 (Build 1177), Powered by wxWidgets 3.1.0, Lua 5.1.*

[69] ”Scripting,” VisionaireWiki, 14.3.2014. [Online]. Saatavissa: http://wiki. visionaire2d.net/index.php?title=Scripting. [Haettu 2.6.2015].

[70] C. Jones, ”Getting Started with AGS - Part 6,” Adventure Game Studio, 17.8.2007. [Online]. Saatavissa: http://www.adventuregamestudio.co.uk/site/ags/ tutorial/ags/6/. [Haettu 17.6.2015].

[71] ”AGS Editor Plugins,” Adventure Game Studio, [Online]. Saatavissa: http:// www.adventuregamestudio.co.uk/site/ags/plugins/editor/. [Haettu 17.6.2015].

[72] ”Graphics View Framework,” The Qt Company, [Online]. Saatavissa: http://doc.qt.io/qt-4.8/graphicsview.html. [Haettu 2.8.2015].

[73] ”Qt,” The Qt Company, [Online]. Saatavissa: http://www.qt.io/.

[74] ”Qt Documentation, Officially Supported Platforms,” The Qt Company, [Online]. Saatavissa: http://doc.qt.io/QtSupportedPlatforms/index.html. [Haettu 17.6.2015].

[75] ”Windows Presentation Foundation,” Microsoft, [Online]. Saatavissa: https:// msdn.microsoft.com/en-us/library/ms754130(v=vs.110).aspx. [Haettu 31.5.2015].

[76] M. Geig, ”Why You Should Be Using the Unity Game Engine,” Pearson Education, Informit, 4.4.2013. [Online]. Saatavissa: http://www.informit.com/ articles/article.aspx?p=2031153. [Haettu 17.6.2015].

[77] ”Plugins (Pro/Mobile-Only Feature),” Unity Technologies, [Online]. Saatavissa: http://docs.unity3d.com/460/Documentation/Manual/Plugins.html. [Haettu 17.6.2015].

[78] ”PRICING,” Unity Technologies, [Online]. Saatavissa: https://store.unity3d.com/products/pricing. [Haettu 17.6.2015].

[79] Atlassian, ”Bitbucket,” [Online]. Saatavissa: https://bitbucket.org/. [Haettu 16.8.2015].

[80] Atlassian, ”SourceTree, A free Git & Mercurial client for Windows or Mac.,” [Online]. Saatavissa: https://www.sourcetreeapp.com/. [Haettu 16.8.2015].

[81] ”Mercurial,” [Online]. Saatavissa: https://mercurial.selenic.com/. [Haettu 16.8.2015].

[82] P. Site, ”PS Site.com: Silence: The Whispered World 2 | Gameplay,” YouTube, 29.1.2015. [Online]. Saatavissa: https://www.youtube.com/watch?v=2787-CVbaok. [Haettu 27.6.2015].

[83] P. Proper, ”Silence: The Whispered World 2 (Gameplay Footage),” YouTube, 4.8.2014. [Online]. Saatavissa: https://www.youtube.com/watch?v=hxkz2mujh9A. [Haettu 27.6.2015].

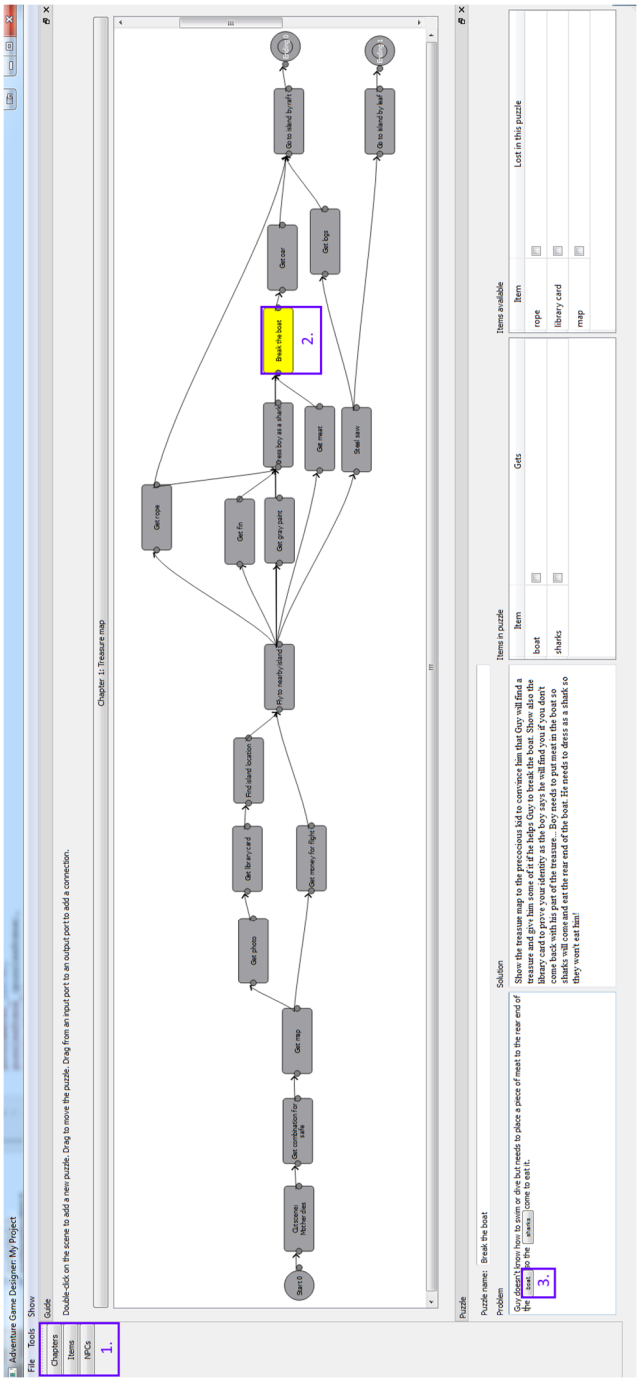
[84] A. Plotkin, ”PlotEx: a tool for exploring puzzle plot constraints,” 1.6.2012. [Online]. Saatavissa: http://eblong.com/zarf/plotex/. [Haettu 24.6.2015].

[85] ”Topic: An Online Adventure Game Design Document App (The Plan),” Adventure Game Studio, forums, 2012. [Online]. Saatavissa: http://www.adventuregamestudio.co.uk/forums/index.php?topic=45944.0. [Haettu 10.1.2015].

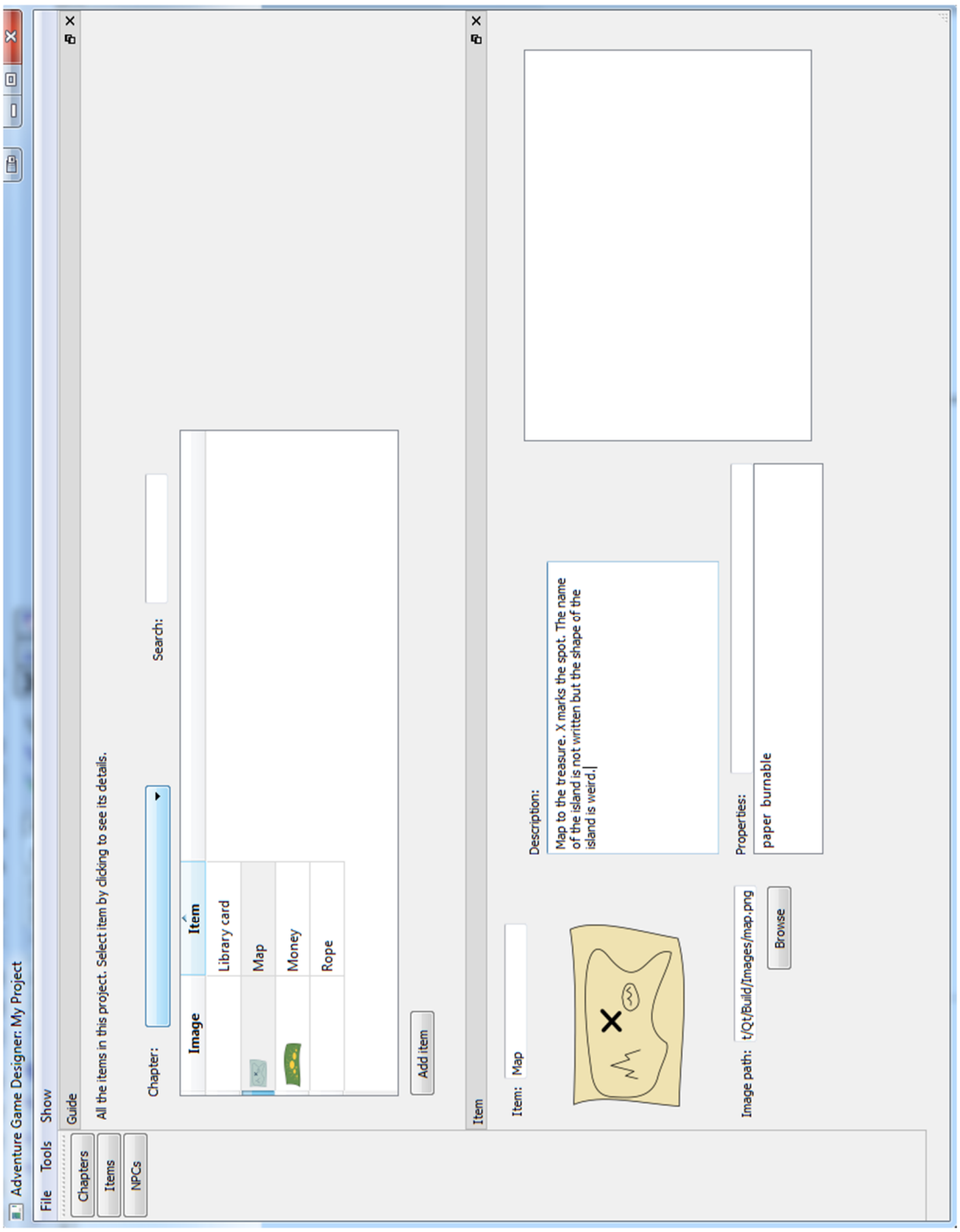
[86] ”Top Project Management Software Products,” Capterra, [Online]. Saatavissa: http://www.capterra.com/project-management-software/. [Haettu 5.7.2015].

LIITE A: NAVIGOINTI PUZZLES-NÄKYMÄSSÄ

Kuvassa on esitetty, miten Puzzles-näkymässä voidaan navigoida. Vasemmassa reunassa (1.) on navigointipalkki, joka on koko ajan näkyvillä. Siitä voidaan navigoida Chapters- ja Items-näkymiin. Kun suunnittelija valitsee tehtävän tehtävägraafista klikkaamalla sitä (2.) sen lisänäkymä esitetään alareunassa. Puzzle-lisänäkymän ongelma- ja ratkaisukentistä saadaan esiin luodun tavaran lisänäkymä (3.).



LIITE B: ITEMS-NÄKYMÄ JA ITEM-LISÄNÄKYMÄ



LIITE C: TYÖKALUN GENEROIMA XML-TIEDOSTO

Alla on työkalun generoima XML-tiedosto testipelin lyhennetystä versiosta. XML:ää on myös typistetty merkinnällä ”...”. Puzzles-elementeille ei ole XML:ssä määritelty yhteyksiä, mutta ne määritellään samalla lailla kuin muut yhteydet, esimerkiksi kuten lukujen yhteydet.

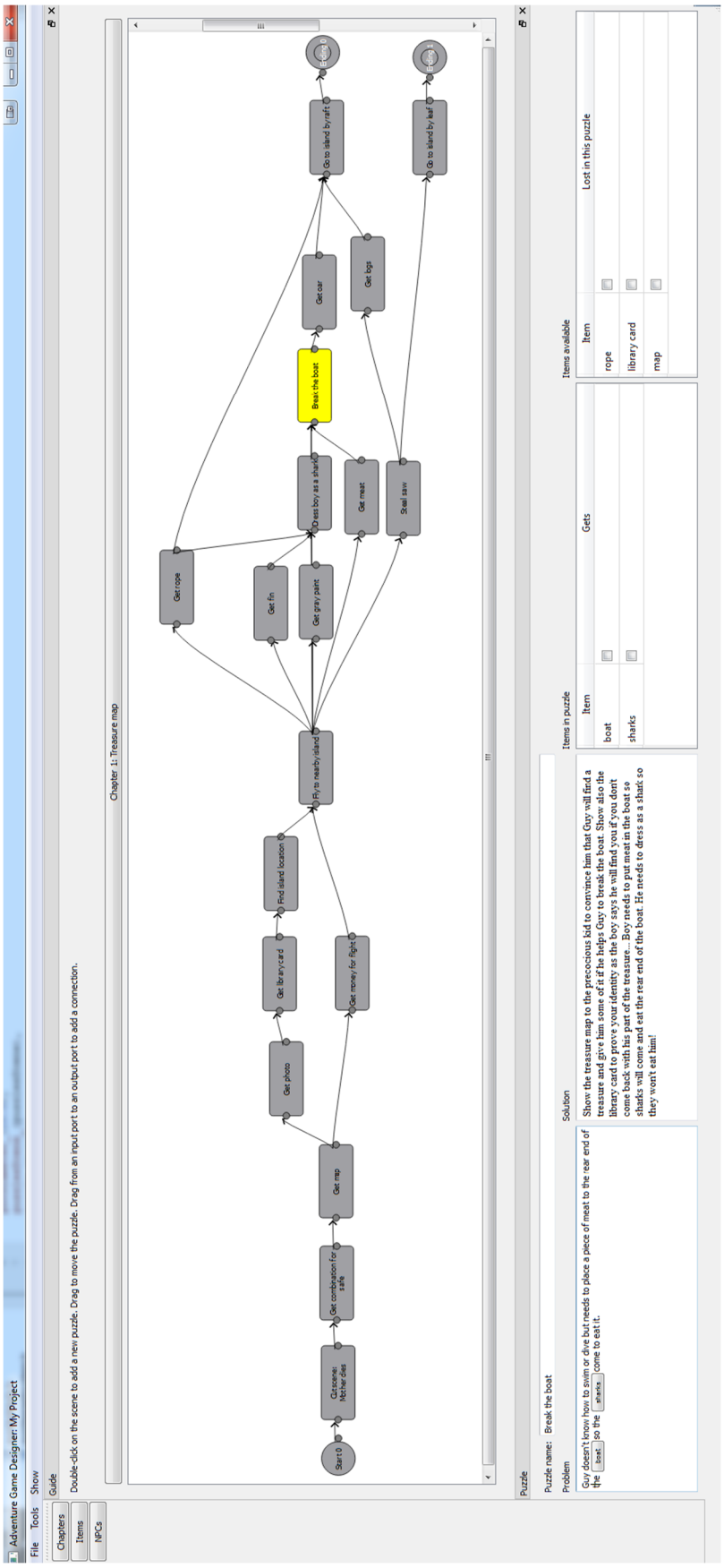
```
<?xml version="1.0" encoding="UTF-8"?>
<chapters>
  <chapter>
    <name>Chapter 1: Treasure map</name>
    <description>Guy inherits a treasure map...</description>
    <starts>
      <start ID="2">
        <name>Start 0</name>
      </start>
    </starts>
    <endings>
      <ending ID="0">
        <name>Ending 0</name>
      </ending>
      <ending ID="8">
        <name>Ending 1</name>
      </ending>
    </endings>
    <puzzlegraph>
      <puzzles>
        <puzzle ID="24">
          <name>Get map</name>
          <problem>The &lt;map&gt; is in a locked safe.</problem>
          <solution>Use combination to open the safe and get map.</solution>
          <items>
            <item CanBeTaken="1">map</item>
          </items>
        </puzzle>
        <puzzle ID="25">
          <name>Find island location</name>
          <problem>Guy needs to check where the island on the treasure map is
                   and how to get there.</problem>
          <solution>Guy gets a &lt;library card&gt; to search the info from
                    the library.</solution>
          <items>
            <item CanBeTaken="1">library card</item>
          </items>
        </puzzle>
        <puzzle ID="26">
          <name>Get money</name>
          <problem>No &lt;money&gt;.</problem>
          <solution>Get money from brother.</solution>
          <items>
            <item CanBeTaken="1">money</item>
          </items>
        </puzzle>
        <puzzle ID="27">
          <name>Fly to nearby island</name>
          <problem>Guy needs to get to the island nearby the treasure
                   island.</problem>
          <solution>Buy tickets to fly to the island.</solution>
          <items/>
        </puzzle>
        <puzzle ID="28">
          <name>Go to island by raft</name>
          <problem></problem>
```

```
      <solution></solution>
      <items/>
    </puzzle>
    <puzzle ID="29">
      <name>Go to island by leaf</name>
      <problem></problem>
      <solution></solution>
      <items/>
    </puzzle>
  </puzzles>
  <connections/>
</puzzlegraph>
<map>
  <rooms>
    <room>
      <name>Bedroom</name>
      <imagepath>D:/omat_ohjelmat/Qt/Build/Images/bedroom.png</imagepath>
      <ids>
        <north>30</north>
        <east>31</east>
        <south>32</south>
        <west>33</west>
      <ids>
    </room>
    <room>
      <name>Hallway</name>
      <imagepath>D:/omat_ohjelmat/Qt/Build/Images/hallway.png</imagepath>
      <ids>
        <north>34</north>
        <east>35</east>
        <south>36</south>
        <west>37</west>
      <ids>
    </room>
    <room>
      <name>Surveillance room</name>
      <imagepath>D:/omat_ohjelmat/Qt/Build/Images/
                surveillanceroom.png</imagepath>
      <ids>
        <north>38</north>
        <east>39</east>
        <south>40</south>
        <west>41</west>
      <ids>
    </room>
    <room>
      <name>Lobby</name>
      <imagepath>D:/omat_ohjelmat/Qt/Build/Images/
                  lobby.png</imagepath>
      <ids>
        <north>42</north>
        <east>43</east>
        <south>44</south>
        <west>45</west>
      <ids>
    </room>
  </rooms>
  <connections>
    <connection>
      <from>31</from>
      <to>37</to>
    </connection>
    <connection>
      <from>37</from>
      <to>31</to>
    </connection>
    <connection>
      <from>36</from>
```

```
        <to>42</to>
      </connection>
      <connection>
        <from>42</from>
        <to>36</to>
      </connection>
      <connection>
        <from>45</from>
        <to>39</to>
      </connection>
      <connection>
        <from>39</from>
        <to>45</to>
      </connection>
    </connections>
  </map>
</chapter>
<chapter>
  <name>Chapter 2.1: Going to the treasure island by raft</name>
  <description>Guy arrives at the left side...</description>
  <starts>
    <start ID="6">
      <name>Start 0</name>
    </start>
  </starts>
  <endings>
    <ending ID="4">
      <name>Ending 0</name>
    </ending>
  </endings>
  ...
</chapter>
<chapter>
  <name>Chapter 2.1: Going to the treasure island by giant leaf</name>
  <description>The giant leaf starts to sink...</description>
  <starts>
    <start ID="12">
      <name>Start 0</name>
    </start>
  </starts>
  <endings>
    <ending ID="10">
      <name>Ending 0</name>
    </ending>
  </endings>
  ...
</chapter>
<chapter>
  <name>Chapter 3: Finding the treasure</name>
  <description>Guy will follow the map...</description>
  <starts>
    <start ID="16">
      <name>Start 0</name>
    </start>
    <start ID="22">
      <name>Start 1</name>
    </start>
  </starts>
  <endings>
    <ending ID="14">
      <name>Ending 0</name>
    </ending>
  </endings>
  ...
</chapter>
<connections>
  <connection>
    <from>0</from>
```

```
      <to>6</to>
    </connection>
    <connection>
      <from>8</from>
      <to>12</to>
    </connection>
    <connection>
      <from>4</from>
      <to>16</to>
    </connection>
    <connection>
      <from>10</from>
      <to>22</to>
    </connection>
  </connections>
</chapters>
<items>
  <item>
    <name>library card</name>
    <imagepath></imagepath>
    <description></description>
    <tags/>
  </item>
  <item>
    <name>map</name>
    <imagepath>D:/omat_ohjelmat/Qt/Build/Images/Map.png</imagepath>
    <description>A drawing of an island with an X mark.</description>
    <tags>
      <tag>paper</tag>
      <tag>burnable</tag>
    </tags>
  </item>
  <item>
    <name>money</name>
    <imagepath></imagepath>
    <description></description>
    <tags/>
  </item>
</items>
```

LIITE D: TESTIPELIN LUVUN 1 TEHTÄVÄGRAAFI



LIITE E: TESTIPELIN KARTTA